View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by ScholarBank@NUS

INTEGRATED VEHICLE DISPATCHING FOR CONTAINER TERMINAL

WANG YUAN

NATIONAL UNIVERSITY OF SINGAPORE

2011

INTEGRATED VEHICLE DISPATCHING FOR CONTAINER TERMINAL

WANG YUAN

(Bachelor of Management, Tianjin University)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2011

Acknowledgements

I would like to express my gratitude to all those who helped me during the writing of this thesis. My deepest gratitude goes first and foremost to A/Prof. LEE Loo Hay, and A/Prof. CHEW Ek Peng, my supervisors, for their consistent encouragement and guidance throughout the whole course of my research. Without their invaluable and illuminating instructions, this thesis would not have reached its present form.

Second, I would like to express my sincere gratitude to A/Prof. POH Kim Leng, and A/Prof. NG Kien Ming, who have referred me to the research engineer position. Gratitude also goes to all other faculty members of the Department of Industrial and Systems Engineering for their kind attention and help in my research.

I also would like to extend my gratitude to all my friends who have made my life in NUS an unforgettable and valuable experience. Thank Hu Qingpei, Han Yongbin, Liu Rujing, Liu Xiao, Xin Yan, Qu Huizhong, Yuan Le, Yang Wei, Wu Yanping, Wang Xiaoying, Wang Qiang, Li Juxin and Celine Neo and all other students in Simulation Lab.

Last, but not the least, I would like to thank my beloved family for their great concern and love all through these years. Special thank is reserved to my boyfriend Dongxiang Zhang for his continuous support and love which has sustained me through the otherwise grueling period of my doctoral study and especially the days I injured my backbone.

Table of Contents

Summary

The dispatching problem in container terminals has received considerable attention from researchers. However, few works have taken into account the coordination among various types of terminal equipment, including Quay Cranes (QC), Prime Movers (PM) and Yard Cranes (YC). To bridge the gap, we address the integrated vehicle dispatching problem in this thesis and design effective models and algorithms to solve the problem.

Firstly, we address an integrated dispatching problem which considers both waiting time at quay side and yard side in a container terminal. In previous works [Kim and Bae (2004), Cheng (2005)], the waiting time at yard side is ignored for simplicity. We argue that this variable plays a significant role in the dispatching problem. To solve the new problem, we build a mixed integer programming (MIP) model. Since existing solvers cannot solve the MIP model in reasonable time, we develop two heuristic algorithms. The first is the variable neighborhood search (VNS) algorithm, which is based on the random exchange of neighborhood, but may terminate with only limited improvement. The second method is based on the combination of genetic algorithm (GA) and the minimum cost flow (MCF) network model. We prove that there exists a set of job ready times in the MCF model which produce the optimal vehicle job sequence. Unlike improving the vehicle job sequence directly in most GA algorithms, we use the job ready times as the chromosome and then use the MCF model to decode the job sequence. This converts the complex MIP model into a simple linear programming (LP) formulation. The experimental results indicate the superiority of the GA-MCF algorithm over the neighborhood search algorithm.

Secondly, we extend the integrated dispatching problem by considering the locations to store the discharging containers. Previous studies simply assume that the yard locations for discharging jobs are given. However, in actual terminal operations, the port operators can also determine the yard location of discharging containers. Thus, we extend the previous problem by considering the storage locations for discharging containers. This has enlarged the solution space for the problem. In order to effectively find a good solution, we use a tree structure to represent the search space and propose three heuristic methods to solve the problem. The three methods are Nested Partition based method (NP), Buffered Semi Greedy method (BSG), and Buffered Probabilistic Greedy method (BPG). Extensive experiments are conducted and the results show that these heuristic methods can find promising solutions in seconds.

Thirdly, we develop an efficient simulation platform to compare and evaluate different dispatching rules to facilitate real time dispatching. In real time dispatching, it is difficult for a port operator to choose a proper rule because the system is highly dynamic and stochastic. The rules might perform differently under different scenarios. In this thesis, we present this simulation platform to evaluate the effectiveness of different rules under different scenarios. This platform not only can work with simple rules, but can also evaluate complex heuristic models which most of the current commercial simulation software would not be able to do so. It can communicate effectively with different solvers which are needed to solve these complicated optimization models.

List of Tables

List of Figures

List of Abbreviations

AGV: Automated Guided Vehicle

ALV: Automated Lifting Vehicle

BSG: Buffered Semi Greedy

BPG: Buffered Probabilistic Greedy

FCFS: First Come First Served

GA: Genetic Algorithm

MCF: Minimum Cost Flow

MILP: Mixed Integer Linear Programming

MIP: Mixed Integer Programming

NP: Nested Partition

PM: Prime Mover

QC: Quay Crane

SA: Simulated Annealing

SC: Straddle Carrier

TEU: Twenty-foot Equivalent Unit

TRACES: Traffic Control Engineering System

YC: Yard Crane

List of Notations

K	the set of QCs
M	the set of PMs
R	the set of YCs
N_k	the number of jobs in the working list of QC k
L	the set of loading jobs
D	the set of discharging jobs
H	the set of all the jobs, $H = L \cup D$
N	total number of container jobs including loading and unloading
(i, α)	the i^{th} job in the sequence list of QC α
(S,D)	Dummy starting job
(E,D)	Dummy ending job
J	the job set which contains all the jobs including dummy starting jobs and dummy ending job, $J = \{(S, D), (E, D)\} \cup H$
J_S	the job set which contains all the jobs including dummy starting jobs, $J_S = \{(S, D)\} \cup H$
J_E	the job set which contains all the jobs including dummy ending job, $J_E = \{(E, D)\} \cup H$
J_r	Set of jobs using yard crane r
C	a huge constant number.
$h_{1(i,\alpha)}$	the QC handling time of job (i, α).

$h_{2(i,\alpha)}$ the YC handling time of job (i,α).

$t_{1(i,\alpha)}$ loaded traveling time of job (i,α) from its origin to its predetermined destination.

$t_{2(i,\alpha)(j,\beta)}$ empty traveling time from destination of job (i,α) to the origin of next job (j,β)

$t^{r}_{1(i,\alpha)}$ loaded traveling time of Discharging job (i,α) from its origin to Yard Crane r

$t^{r}_{2(i,\alpha)(j,\beta)}$ empty traveling time from the destination r of discharging job (i,α) to the origin of next job (j,β)

$X^{m}_{(i,\alpha)(j,\beta)}$ $X^{m}_{(i,\alpha)(j,\beta)}$=1 when PM m moves to the origin of next job (j,β) after just finishing job (i,α); Otherwise $X^{m}_{(i,\alpha)(j,\beta)} = 0$

$Z^{r}_{(i,\alpha)(j,\beta)}$ $Z^{r}_{(i,\alpha)(j,\beta)}$ =1 when the job (j,β) is processed by the same YC r immediately after job (i,α); Otherwise, $Z^{r}_{(i,\alpha)(j,\beta)} = 0$

$T_{1(i,\alpha)}$ The starting time for processing container job (i,α) by QC α

$T_{2(i,\alpha)}$ The starting time for processing container job (i,α) by corresponding YC

$W_{(i,\alpha)r}$ $W_{(i,\alpha)r}$ =1 when job (i,α) is assigned to yard location r;

Otherwise $W_{(i,\alpha)r} = 0$

1 Introduction and Overview

1.1 Introduction

Maritime transport remains the strong backbone supporting globalization as 80 percent of international trade by volume is transported via sea. With world trade booming, competition between major seaports is becoming intense. Hence it is important for port operators to develop different decision tools and optimization algorithms so as to improve its performance and increase its competitiveness.

To increase productivity of the terminal, it is necessary to coordinate different terminal equipment to ensure a seamless flow of containers within the terminal. A schematic diagram of the typical processes in a container terminal is shown in Figure 1.1 (Vis et al. 2003). Container activities can be categorized into three types: import, export, and transshipment activities. For export activities, the containers are brought in by shippers and will be stored at their designated locations in the storage yard. When it is time to load the containers, they are retrieved from the stored locations by yard cranes (YC) and transported by man-driven vehicles (Prime Movers, PM) to the quay side. The quay cranes (QC) then remove the containers from the vehicles and load them onto the vessels. The processes for import activities are performed similarly except that they are done in the reverse order. For transshipment activities, the processes are slightly different. The containers will be stored in the storage yard after they are unloaded from the vessel, and will be finally loaded onto other vessels.

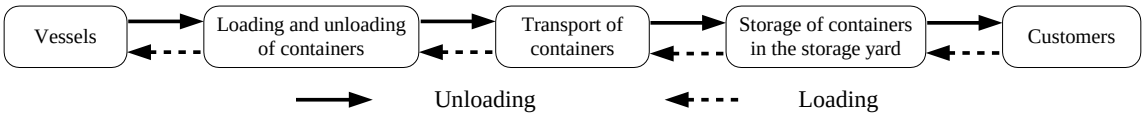

Figure 1.1 A flow diagram demonstrating the interaction between terminal processes

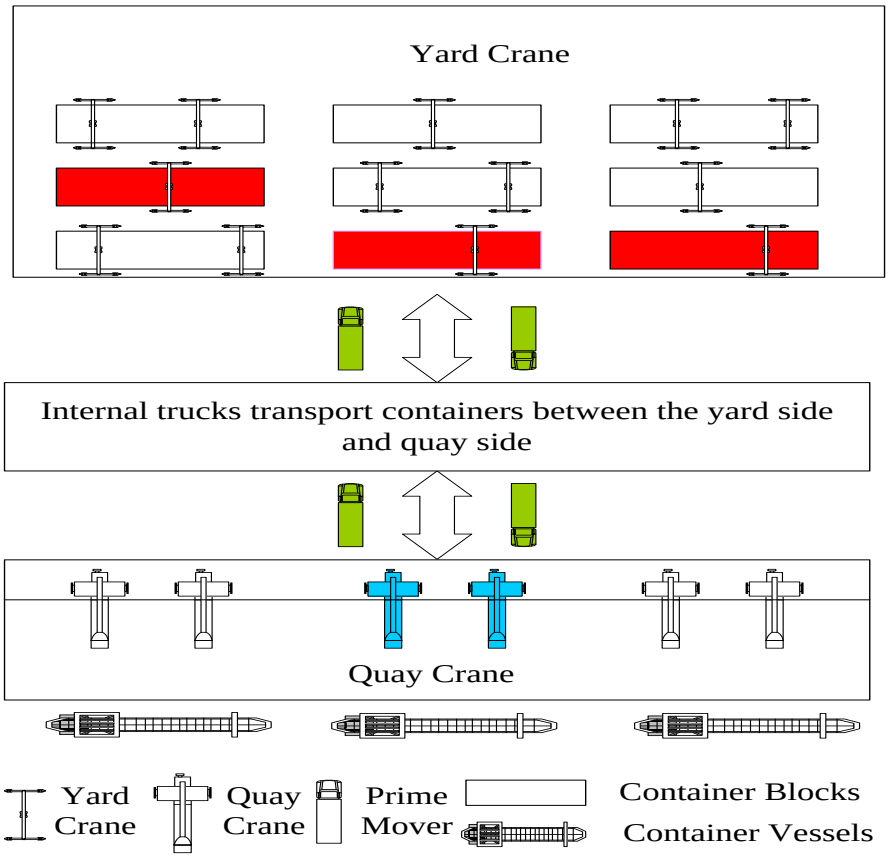

Figure 1.2 Working flow of container dispatching problem

Figure1.2 shows working flow of container dispatching problem. When a vessel arrives at the berth, QCs discharge the import and transshipment containers from the ship onto PMs to transport them to storage locations at the yard side. At the yard side, YCs unload these containers from the PMs to the designated storage locations. A similar process can be found for loading the export and transshipment containers from the yard side to the ship.

The transportation between the quay side and the yard side plays a crucial role in determining the productivity of the terminal, because it might delay the QC or YC operations if PMs do not arrive in time or may cause traffic congestion if they arrive too early. In practice, since the coordination between the various types of equipment at the operational level is complex due to the traffic congestion, traffic delay, equipment breakdown and other dynamic uncertainties in the port, optimizing this dispatching problem becomes very challenging. Therefore, a decision supporting system is needed to provide fast and intelligent solutions to guide the port operators in dispatching and routing vehicles. This study aims to address dispatching related problems as follows by developing algorithms to improve the port operation performance.

1.1.1 Integrated Vehicle Dispatching Problem

In the past, most of previous studies on terminal vehicle dispatching problem do not consider the delay at the yard side and they usually assume that YC is always available when needed. This assumption works only when the yard is not congested, or there are enough YCs. However, this might not be the case especially when there is high traffic volume. Furthermore, most of these works confine their works to gate ports where the jobs are usually either import or export jobs. In this thesis, we consider the delay at the yard side, and also include both the loading and discharging jobs simultaneously. Moreover, the PMs are pooled among all the QCs rather than dedicated to a certain QC. This thesis seeks to provide an efficient way of dispatching vehicles to minimize the makespan time at the quay side for a given number of container jobs by considering all equipment.

We build a Mixed Integer Programming (MIP) model to solve this integrated dispatching problem. However, the existing optimization solver cannot solve this MIP model when the problem size increases dramatically. Thus we develop two heuristic methods to solve the problem, that can improve the initial solution obtained from the Minimum Cost Flow (MCF) model which ignores the yard side waiting time. The first method is a neighborhood search method named Variable Neighborhood Search (VNS) method. The second method is based on Genetic Algorithm (GA). Before implementing GA, we prove that there exists a property in the Minimum Cost Flow (MCF) model that we can obtain the optimal job sequence when setting the proper ready times in MCF. Thus we implement GA by using this special property. Unlike the common Genetic Algorithm (GA) which usually represents the chromosome using job sequence, we use the ready time for jobs as the representation of the chromosome, and the MCF model is then used to decode the chromosome to determine the job sequence for prime movers. The experiment results indicate the superiority of the GA-MCF based algorithm over the neighborhood search algorithm.

1.1.2 Container Dispatching and Location Problem

Previous dispatching studies only focus on optimizing job sequence on each vehicle while assuming that yard location for the discharging container is known. However, it might be sub-optimal if we focus merely on planning vehicle dispatching without considering the yard locations for the inbound container. The storage location for the inbound job is also an important decision variable which may affect the performance of the whole system. Unlike previous works which only consider how to dispatch vehicles to improve QC

productivity, we investigate in this thesis the whole dispatching process in both QC and YC performances so as to improve the productivity rate of the whole terminal.

To investigate this integrated problem, we present three heuristic methods: Nested Partition (NP) based method, Buffered Semi Greedy method (BSG), and Buffered Probabilistic Greedy method (BPG). They all use tree structure to find the job sequence as well as the storage locations for the inbound containers. However, the NP method spends too much time on sampling and selection since it needs to calculate the completed solution in each stage. On the contrary, BSG and BPG methods save computing budget since they only calculate the current partial solution in each stage, and BPG can capture current information to determine the next searching direction in a probabilistic manner. We also propose a new measure to evaluate the performance of these heuristic algorithms, which considers the tradeoff between the elapsed time in exploring the solution space and the quality of the result. Extensive experiments are conducted and results show that these proposed heuristic methods can find a promising solution in reasonable time.

1.1.3 The Integrated Simulation Platform for Real Time Dispatching

For real time dispatching, we need to make decisions based on the most updated information, which usually cannot be considered directly during the planning phase. In practice, port planners usually use simple greedy rules to do real time dispatching under certain scenarios. Simulation is widely used here to evaluate the effectiveness of these simple rules. On the other hand, some researchers propose more complicated rules which can capture more information by using look ahead planning. It is not easy to know the effectiveness of these proposed rules, since coding these complicated rules into current

commercial simulation software is difficult and time consuming. In addition, it is not easy to choose a proper rule between a simple rule and a complex one under different scenarios. Therefore, we develop a useful simulation platform to aid terminal operators in implementing and evaluating different rules efficiently. This platform is powerful not only in working with simple rules, but also in evaluating complex heuristic models.

By using this simulation platform, it becomes quite flexible in testing different rules under different simulation scenarios. Besides, the simulation and the external optimization model can communicate with each other to share current state information. With this simulation platform, we can evaluate the performance of different simple or even complicated models in real time environment.

1.2 Contribution of the Thesis

In summary, this thesis addresses the integrated dispatching problem which takes into account the coordination among all equipment for a container terminal. The contributions of this thesis are listed as follows:

- Different from traditional container dispatching problem, we present a new problem which seeks to solve a vehicle dispatching problem for both the discharging and loading containers by taking into account waiting times at the quay side and the yard side in a container terminal. However, most of the existing works either ignore yard side waiting time or only focus on a single type of container. The complexity of this integrated problem makes it challenging to solve

under any commercial optimization software but we manage to propose efficient algorithms to provide good solutions.

- In terms of methodology, this thesis aims to define a problem structure where a more effective searching algorithm can be developed. In the first dispatching problem, we use a good neighborhood structure to explore the solution space. Instead of solving the complex MIP model directly, we propose solving a simple MCF model. We prove that there exists a set of MCF parameters which can result in the optimal vehicle job sequences. To find this set of MCF parameters, we propose using GA. Unlike the typical GA which usually represents the chromosome using the job sequence, we use the job ready time in MCF as the representation of the chromosome, and the MCF model is then used to decode the chromosome to determine the job sequence. Due to the fact that the job ready time is continuous in nature and has good neighborhood structure, GA is able to give us good results.
- In the dispatching and location problem, we present the tree representation for solution structure and propose three efficient algorithms. In each algorithm, we learn the information obtained from every exploration stage to determine the next searching direction. These efficient tree based searching approaches can obtain better solutions in a very fast speed. Moreover, we propose a comparison mechanism which considers the ratio of solution quality and CPU time.
- We develop a simulation platform which can be used to evaluate and compare different dispatching methods from simple rules to complicated models for real

time dispatching. Unlike other simulation software, this platform not only works with simple rules, but also can help in evaluating complex heuristic algorithms.

1.3 Organization of the Thesis

This thesis consists of six chapters. The rest of this thesis is organized as follows:

Chapter 2 reviews related literature on port operations including capacity planning, berth allocation, quay crane assignment, ship stowage, yard configuration, yard allocation, yard crane deployment, prime mover deployment, and integrated terminal study, etc.

In Chapter 3, the formulated mixed integer linear programming model for the vehicle dispatching problem is presented, in which waiting time at both quay side and yard side in a container terminal is considered. Two heuristics to solve this model as well as experimental results are presented.

Chapter 4 describes the relaxed yard location for discharged container model and the proposed heuristics for solving the model. Numerical experiments on the relaxed yard location problem are conducted and computational results are presented in this chapter.

In Chapter 5, the real time concept for vehicle dispatching is studied by simulation platform. Two serials of experiments to evaluate this framework for real time dispatching are conducted. One is to analyze the performance of simple dispatching rules and strategies embodied directly in the simulation platform; the other is to demonstrate the performance of complicated dispatching optimization models via this framework.

Finally, in Chapter 6, we consolidate the findings from previous chapters and discuss some issues for future research.

2 Literature Review

There are voluminous research works in the area of container terminal operations. Hence, it is useful to provide a short classification review to introduce the related topics that people have done. In this chapter, we summarize and categorize the operations in container terminals. Our main focus is on the transportation optimization which is most related with our work. Literature reviews on port operations can also be found in Vis and de Koster (2003), Steenken et al. (2004) and Robert Stahlbock & Stefan (2008).

2.1 Ship Planning Process

When a ship arrives at the terminal, it has to find a place to moor. The berth (place for ship to moor) together with several quay cranes will be assigned to the ship. Ship planning consists of three partial processes: the berth allocation planning, the stowage planning and the quay crane scheduling.

2.1.1 Berth Allocation Problem

Imai et al. (1997) study the problem of optimally allocating berths to ships while minimizing the sum of ship turnaround times and minimizing dissatisfaction of the ship owners in terms of the berthing order. The problem is then reduced to a single objective problem which becomes similar to the classical assignment problem. Imai et al. (2001) propose a heuristic procedure based on Lagrangian relaxation to solve the dynamic berth allocation problem. Numerical experiments show that the proposed algorithm is adaptable for real world applications. Nishimura et al. (2001) propose a genetic algorithm based

heuristic to investigate the berth allocation problem in the public berth system. Guan et al. (2002) consider the berth allocation problem as a scheduling problem and conduct the worst case analysis. Imai et al. (2003) propose a genetic algorithm based heuristic to solve the berth allocation problem with service priority. Kim and Moon (2003) present a mixed integer programming model and use simulated annealing algorithm to determine the berthing times and positions of vessels. Park and Kim (2003) investigate the berth allocation problem together with the quay crane scheduling problem. A two-phase solution procedure is suggested to solve the formulated model. Guan and Cheung (2004) investigate the berth allocation problem by using the tree search procedure. The objective of this study is to minimize the total weighted flow time. Cordeau et al. (2005) propose a tabu search algorithm for solving the berth allocation problem and quay crane allocation problem sequentially. Moorthy and Teo (2006) model the berth allocation problem as a rectangle packing problem on a cylinder and use a sequence pair based simulated annealing algorithm to solve the problem. In Cordeau et al. (2007), the berth allocation problem is formulated as a generalized quadratic assignment problem with the objective of minimizing the sum of assignment and traffic costs. In Bae et al. (2007), a dynamic berth scheduling method is proposed with the objective of minimizing the travel costs of vehicles during the ship operation, the tardiness costs, the earliness costs as well as the costs for a vessel's waiting time.

For additional references dealing with the berth allocation problem, one can refer to Park and Kim (2002), Imai et al (2006a, 2007b).

2.1.2 Stowage Planning Problem

Avriel and Penn (1993) study the ship stowage problem to minimize the shifting of containers on board without considering the stability constraints. Avriel et al. (2000) investigate the same problem based on the model in (Avriel and Penn 1993, 1999). They find a relationship between the ship stowage problem and the coloring of circle graphs problem. Consequently, they improve Unger's upper bound on the coloring number of circle graphs to solve the ship stowage problem. Wilson and Roach (1999) propose a tabu-search based heuristic algorithm to solve the ship stowage problem. Wilson and Roach (2000) propose a methodology for the automatic generation of computerized solutions to the ship stowage problem. The methodology progressively refines the placement of containers within the cargo-space of a container ship and can generate good, if not optimal, solutions for the problem within a reasonable time. Dubrovsky et al. (2002) use a genetic algorithm based heuristic to solve the ship stowage problem. An efficient encoding of solutions is proposed to reduce the search space. Extensive simulation runs show the efficiency of the encoding of solutions. The encoding can also be incorporated with the ship stability constraints. Kang and Kim (2002) investigate the ship stowage problem to minimize the shifting time, reduce quay crane movements, and maintain the stability of the ship. A solution procedure divides the problem into two sub-problems, each of which is solved iteratively using information from the other. Ambrosino et al. (2006) investigate the stowage planning problem by presenting a three stage algorithm with the objective of minimizinig the total loading time. Imai et al. (2006b) study container stowage and loading plans of a ship by proposing a multi objective model to consider the minimum number of container re-handles in yard as well as ship stability. Additional references

dealing with the ship stowage problem are, e.g., Steenken et al. (2001), Wilson et al. (2001), Roach and Wilson (2002), Giemsch and Jellinghaus (2003), and Álvarez (2006, 2007).

2.1.3 Quay Crane Scheduling Problem

Crane scheduling is referred to the allocation of quay cranes to ships and the ships' sections. Daganzo (1989) uses a simple static case and dynamic case with berth length limitations to study the quay crane scheduling problem. The objective of the study is to minimize the turnaround time of all the ships. Two methods are proposed: one is exact method and the other is approximation one for implementation purpose. In Peterkofsky and Daganzo (1990), the quay crane scheduling problem is considered as an "open shop" problem with parallel identical machines. And a branch and bound method is proposed to with the objective of minimizing the ship delay costs. Kim and Park (2004) propose a branch and bound algorithm and a greedy randomized adaptive search procedure (GRASP) based heuristic to solve the quay crane scheduling problem with the objective of minimizing the weighted sum of the makespan time at quay side. Canonaco et al. (2007) present a queuing network model to solve this problem with the objective of minimizing the vessel's turnaround time. Liang and Mi(2007) propose a multi-objective model for the quay crane scheduling problem. The proposed model seeks to minimize the service and delay time of the vessel, as well as the standard deviation of the quay cranes' working time. Linn et al. (2007) study this problem by proposing a machine learning algorithm based on artificial neural network paradigm. Additional references dealing with the quay crane scheduling are, e.g., Zaffalon et al. (1998) and Murty et al (2006). Quay crane with

twin lift mode is also considered in the quay crane scheduling problem in Johansen (2006) and Shanghai Zhenhua Port Machinery (2007). Some studies considering dual cycling of cranes in quay crane scheduling problem are, Goodchild (2005, 2006) and Goodchild and Daganzo (2004, 2005, 2006, 2007).

2.2 Container Yard Storage Problem

A container's position in the storage area is defined by the block, the bay, the row and the tier. In yard planning systems, stack areas and storage capacities are allocated to a ship's arrival in advance according to the number of import and export containers expected. The efficiency of stacking depends greatly on the strategies of allocating storage space to arriving containers. In the early stage of yard storage planning study, lots of works are focused on eliminating the unproductive reshuffles. Simulation is a useful tool in early studies, such as Chung et al. (1988) and Sculli and Hui (1988). Kim (1997) proposes a methodology to estimate the expected number of reshuffles to pick up an arbitrary container and the total number of reshuffles to pick up all the containers in a block for a given initial stacking configuration. He finds that the height and width of the container block are the key factors. Gambardella et al. (1998) propose an integer linear programming model for the storage allocation problem and use simulation to validate the robustness of the model. Kim and Bae (1998) discuss how to reshuffle export containers in container terminals. Kim and Kim (1999a) study the import container allocation problem where the arrival rate of import containers is constant, cyclic, and dynamic. Kim et al. (2000) propose a methodology to determine the storage location of an arriving export container by considering its weight. Preston and Kozan (2001) develop a genetic

algorithm based heuristic for container allocation problem with the objective of minimizing the turnaround time of container vessels. In Kim and Park (2003) the storage space allocation problem for export containers is studied. A mixed integer linear programming model is formulated for the transfer system. Zhang et al. (2003) study the storage space allocation problem in the storage yard by a two stage rolling-horizon approach. At the first stage, the total number of containers to be stored in each container block in each shift is determined to balance the workload among blocks. Based on the result of the first stage problem, the number of containers associated with each vessel is determined to minimize the total traveling distance at the second stage. Murty et al. (2005) develop an online dispatching procedure for assigning containers to storage locations. To reduce traffic congestion of prime movers, a fill ratio equalization approach is used to allocate containers to the storage locations. Dekker et al. (2006) study the storage allocation problem via simulation for an automated container terminal. Several variants of consignment strategy, in which the same group of containers are stored together, are discussed. Lee et al. (2006) study the storage allocation problem in transshipment hubs. The consignment strategy, in which containers to the same destination vessel are stored in the same sub-blocks, is used to reduce the number of reshuffles. A high-low workload balancing protocol is used to reduce traffic congestion of prime movers. Hirashima et al. (2006) propose a Q-learning algorithm to solve export container allocation problem with the objective of minimizing vessel's turnaround time. Kang et al. (2006a, b) propose a Simulated Annealing (SA) based heuristic to study this problem for export containers. The results show that using machine learning algorithm can lead to better solutions. Kozan and Preston (2006) present an iterative search algorithm for the integrated container-transfer and container-allocation model to determine the optimal storage strategy and

corresponding handling schedule. Kim and Kim (2007) investigate the yard storage problem by proposing a cost model to encourage short time storage time in the yard.

For additional reference, one can refer to Castilho and Daganzo (1993), Taleb-Ibrahimi et al. (1993), Holgu ń-Versa and Jara-D áz (1996), Chen (1999, 2000), Kozan and Preston (1999), Lim and Xu (2006) and Kim et al. (2007).

2.3 Vehicle Planning Problem

For port container terminals, one of the decisions is the determination of the necessary number of transport vehicles. Steenken (1992) develops a linear assignment model to determine the number of straddle carriers in a container terminal. Vis et al. (2001) develop a minimum flow algorithm to determine the necessary number of automated guided vehicles required in a semi-automated container terminal. Koo et al. (2004) investigate the fleet size problem to determine the necessary number of vehicles required to handle the containers. In this thesis, we focus on the vehicle dispatching problem. Hence in the following section, we mainly discuss vehicle transportation optimization.

2.3.1 Vehicle Dispatching Problem

A decision at the operational level is to determine which vehicle transports which container. For conventional trucks and trailer systems, the vehicle dispatching problem is widely studied. Bish et al. (2001) focus on the NP hard problem of dispatching vehicles and assigning a yard location to each discharging container in order to minimize the makespan. A heuristic algorithm is proposed to solve the formulated assignment model.

The effectiveness of the heuristic algorithm is analyzed from both the worst-case and computational points of view. Narasimhan and Palekar (2002) study the vehicle routing problem to minimize the time taken to load the containers from the storage yard onto the vessel. An integer programming model is formulated and a branch-and-bound based enumerative method is developed to solve the model. Computational experiments are conducted to evaluate the heuristic algorithm. Bish (2003) proposes a heuristic method to solve the problem of dispatching vehicles to containers, determining storage location for each discharging container while scheduling loading and discharging operations on quay cranes. In Bish et al. (2005), an extension of the problem in Bish (2003) is studied. They develop easily implementable heuristic algorithms for this problem and identify the absolute and asymptotic worst-case performance ratios of the proposed heuristics. Numerical experiments show that these heuristics can generate near-optimal or optimal solutions for simple or general setting scenarios. Li and Vairaktarakis (2004) investigate the problem of optimizing the time for loading and unloading containers to and from a ship in a container terminal. An optimal algorithm and some efficient heuristics are developed to solve the problem. The effectiveness of the heuristics is studied both analytically and computationally. Ng and Mak (2004) develop an algorithm to sequence trucks to enter the working lane for export containers. The objective of this study is to reduce congestion of the working lanes. The dynamic trailer routing problem is discussed in Nishimura et al. (2005). In this study the yard trailers are assigned to specific quay cranes and the capacity of the vehicles can be one (single trailer) or two (multi-trailer). Zhang et al. (2005) present three MIP models for vehicle dispatching problem in a container terminal in which the starting times of jobs as well as the work sequence of vehicles need to be determined. The models only consider the unloading phase of a vessel

in one berth and vehicles are assumed to be dedicated to a certain quay crane. Ng et al. (2007) study the problem of scheduling a fleet of trucks to perform a set of transportation jobs in a container terminal using a genetic algorithm. They focus on the scheduling the job order of trucks to minimize makespan. In Chen et al. (2007), the dispatching problem is formulated as a hybrid flow shop scheduling problem with precedence and blocking constraints. A tabu search based algorithm is developed to solve this problem. The results show that a good initial solution is important.

2.3.2 SC and ALV Dispatching Problem

Straddle carriers are alternative vehicles for the transport, retrieval and storage of containers. Thus the routing of straddle carriers has received much attention from the researchers. In Steenken (1992) the routing problem of straddle carriers is studied. The problem is formulated as a linear assignment problem with the objective of minimizing the empty-travel distance by combining loading and unloading jobs. As a result, a saving of 13% in the empty travel distance is obtained. In Steenken et al. (1993), a network problem with minimum cost is formulated to determine the route of straddle carriers. A saving of 20-35% is obtained in the empty travel distance. Kim and Kim (1999c) study the single straddle carrier routing problem to load export containers onto a containership. An integer programming model is developed with the objective of minimizing the total traveling time of the straddle carrier. An efficient algorithm for the integer programming model is proposed. Kim and Kim (1999b) study the straddle carrier routing problem during the loading operation of export containers. The objective is to minimize the total traveling distance of all the straddle carriers in the storage yard. A beam search algorithm

is developed to solve the straddle carrier routing problem. Numerical experiments are carried out to evaluate the proposed algorithm. Böse et al. (2000) study the straddle carrier routing problem between the vessel and the storage yard. The objective for the study is to minimize the vessel turnaround time and to maximize the productivity of the yard cranes. They investigate different dispatching strategies for straddle carriers. An evolutionary algorithm is proposed to improve the solution quality.

Automated Lifting Vehicle (ALV) has the capability to lift a container from the ground. Nguyen and Kim (2007) discuss the dispatching of ALVs. They propose a heuristic algorithm based on the multiple traveling salesman problem. One important assumption in their work is that the waiting time at yard cranes is negligible since automated yard crane is not a bottleneck.

2.3.3 AGV Dispatching Problem

Recently, more container terminals utilize automated transporters, like AGVs. Therefore the research on the dispatching of AGVs becomes important. Evers and Koppers (1996) develop a formal tool to control the large number of AGVs. Simulation models are built to evaluate the various dispatching rules. In Chen (1998) an effective dispatching rule is developed for assigning AGVs to containers. A greedy algorithm is proposed to solve the problem. Simulation shows the solution obtained from the proposed algorithm is near optimal. In Duinkerken et al. (1999), a control system called TRACES (Traffic Control Engineering System) is presented to coordinate the traffic flow of AGVs. A prototype is built as a pilot-study for the TRACES system based on an existing automated container terminal. Kim and Bae (1999) formulate a mixed integer linear programming model for

dispatching AGVs with the objective of minimizing the delays of the containerships and the traveling time of the AGVs. Gademann and van de Velde (2000) determine the home positions for AGVs in a loop layout. A home position is the location where AGVs will park if they are idle. The home position should be selected so as to minimize the total response time for the AGVs. Reveliotis (2000) models a conflict-free AGV system, in which a new conflict resolution strategy is proposed. The strategy employs zone control to determine vehicle routes incrementally. In Bish et al. (2001) an extension of the problem in Chen (1998) is studied. They consider the problem of dispatching AGVs to containers and the container storage problem at the same time. A heuristic method is proposed to solve the problem with the objective of minimizing the unloading time of the containers. In Van der Meer (2000), the control of AGVs is studied in the automated container terminals. Chan (2001) develops a network flow model to dispatch AGVs to containers. Several heuristic algorithms are proposed and tested in the case of single load for each AGV. Computational results show that the proposed dispatching strategy outperforms the current dispatching strategy. Lim et al. (2003) suggest using an auction algorithm for the AGV dispatching problem. Different from traditional dispatching rules, the proposed dispatching rule looks into the future for an efficient assignment of delivery tasks to vehicles and also multiple tasks are matched with multiple vehicles. Moorthy et al. (2003) propose an efficient AGV deadlock prediction and avoidance algorithm for a large-scale container terminal. An AutoMod simulation model is developed to evaluate the algorithm. Simulation results show that potential deadlock situations can be detected and avoided by the algorithm proposed. In Grunow et al. (2004), a multi-load AGV dispatching problem is studied. A flexible priority rule based approach is developed for an online logistics control system. A mixed integer linear programming (MILP) model is formulated to

evaluate the online policy. The performance of the priority rule and the MILP model are analyzed for several scenarios. Lehmann et al. (2006) investigate the deadlock handling of AGVs for an automated terminal. Two methods to detect the deadlock are discussed. One is based on the matrix representation of the terminal system. The other directly traces the requests for each individual resource. Briskon and Hartmann (2006) and Briskorn et al. (2006) investigate the problem to assign transportation jobs to AGVs within a terminal control system in real time. They present an inventory-based formulation for the assignment problem to avoid the estimation of driving times, completion times, and due times. Duinkerken et al. (2006) conduct simulation to compare different trajectory planning of AGVs. The experiments show that the AGV's free ranging capacity results in better solution in a dynamic approach. Grunow et al. (2006) present a simulation study of AGV dispatching strategies in an automated container terminal. The dual load mode is used in the study. The performance of the proposed dispatching strategies is evaluated using a scalable simulation model. Simulation results show that the proposed off-line heuristic strategy outperforms the existing on-line strategies.

Additional references dealing with the AGV dispatching problem are, e.g., Zaffalon et al. (1998), van der Meer (2000), Leong (2001), Schneidereit (2003), Liu et al. (2004), Nishimura et al. (2005),Vis et al. (2006) and Lau er al. (2007).

2.4 Integrated Study for Terminal Planning

In previous sections, most of the works aim to solve isolated terminal operation problems. However it is necessary to study the container terminal as a whole system by integration

of various operations connected to each other. In this section, we review studies regarding integrative views on container operation optimization. These studies can be divided into simulation approach and analytical approach.

2.4.1 Simulation Approach

Most of the researchers use simulation as the methodology to study a whole terminal. Gibson et al. (1992) develop a comprehensive simulation model based on a traditional queuing model that can track individual vehicles and provide system wide performance measures. Koh et al. (1996) use a simulation model to preview the integrated plan of a container port operation. Simulation results give port operators an opportunity to see the performance of the proposed plan and make changes to them before committing them to operations. Charnes et al. (1996) consider the priority of containers via simulation. The current trend of service differentiation is also presented. The conceptual and computational characteristics of the simulation system are described together with the calibration process. Konings (1996) proposes the concept of "integrated center for the transshipment, storage, collection, and distribution of goods". The integrated center is characterized by the spatial and functional integration of container handling and storage. Kozan (1997b) conducts a comparison between the analytical and simulation planning models for a whole container terminal. Gambardella et al. (1998) analyze the resource allocation problem via a simulation model. Bruzzone et.al (1999) show the benefit and effectiveness of the simulation approach for managing complex container ports. Yun and Choi(1999) analyze the performance of a container terminal system in Pusan by an object-oriented simulation model. Duinkerken et al. (2000, 2001) develop an integrated

simulation model which is generic and configurable. Different stacking policies are implemented and tested in the model. Duinkerken et al. (2002) study the same problem as Duinkerken et al. (2001) in which the quay transport system using AGVs is used to deduce the logistic principles for the design of a terminal layout and operational control. Two new designs called "circulation layout" and "crossover layout" are tested by simulation experiments. They conclude that the "crossover layout" is better and requires less AGVs. Kia et al. (2002) conduct a comparison between two different operational systems (current and proposed) statistically via a simulation model and propose an operational method to reduce port terminal congestion and increase the capacity of terminal. Nam et.al. (2002) determine the optimal number of berths and quay crane assignment in Pusan. Shabayek and Yeung (2002) analyze the performance of the operations in a terminal in Hong Kong via simulation. Liu et al. (2002) design, analyze and evaluate four different automated container terminal concepts including automated guided vehicles, linear motor conveyance system, overhead grid rail system, and automated storage and retrieval system. Nevins et al. (1998a, 1998b) develop a seaport simulation model that computes throughput capability and determines resource utilization at a high level of detail. The simulation allows for multiple cargo types as well as multiple ship types. Rebollo et al. (2000) present a multi-agent system to simulate the port container terminal management, and found solutions for the automatic container allocation problem. Demirci (2003) uses simulation to find that the most critical bottleneck points are created by loading/unloading vehicles and an investment strategy is applied to the model for load balancing of the port. Lee et al. (2003) develop a simulation model for port operations to model a supply-chain network in quantity approach and to evaluate its supply-chain performance based on proposed strategies. Sgouridis et al. (2003) focus on

the simulation of incoming containers transported on trucks. Liu et al. (2004) use simulation models to analyze the performance of terminal automation and layout. Hartmann (2004) introduces an approach to generate scenarios of a container terminal, which can be used as input of simulation models and optimization problems. Yang et al. (2004) use simulation to analyze the effect of increasing the number of Automated Lifting Vehicles on the productivity of the terminal. Duinkerken et al. (2006) develop a simulation model combined with a rule-based control system to compare and evaluate different container transportation means. Bielli et al. (2006) elaborate an object oriented design of a simulator for container terminals. Every equipment, queue, area is implemented as an object and communications among objects are implemented as messages. Ottjes et al. (2006) introduce a generic simulation model structure for the design and evaluation of multi-terminal systems. Henesey et al. (2002, 2003a, 2003b, 2009) and Henesey (2006) employed a multi-agent based simulation approach for the evaluation of container terminal management operations. The approach aims to at planning and coordinating the processes within the terminal by mapping the terminal's objects and resources. Franz et al. (2007) presented a market-based approach for integrated container terminal management including specific market-mechanisms as well as a prototypical multi-agent based simulator. Lee et al. (2007, 2008) develop simulation models to investigate the impact of different vehicles and different yard layouts on port operations. They further build a program named Automated Layout Generation to generate different simulation models. Ha et al. (2007) provide a 3D real-time-visualization simulation model which depicts terminal equipment behaviors in details. This model is useful for assessment of the performance of prospective new equipment. Petering (2007) develope a comprehensive simulation model to address issues in terminal design, storage

and retrieval location and yard crane control. Hadjiconstantinou and Ma (2009) develop a decision support system to optimize yard operations by considering all container flows through the yard and used a simulation model for validation. However none of these studies can provide flexible simulation platform for evaluating different planning methods. The objective of this thesis is to study the terminal traffic coordination with different level of sophistication.

2.4.2 Analytical Approach

There are also other methodologies to study the container terminal as a whole system. Bish et al. (2001) focus on the NP hard problem of dispatching vehicles and assigning a yard location to each discharging container in order to minimize the makespan. Bish (2003) proposes a heuristic method to solve the problem of dispatching vehicles to containers, determining storage location for each discharging container while scheduling loading and discharging operations on quay cranes. However, one important assumption in their studies is that they did not consider yard side waiting time. Meersmans and Wagelmans (2001a, 2001b) investigate the integrated scheduling of various types of handling equipment in automated container terminals by using a branch and bound algorithm. The overall objective is to minimize the makespan of the scheduling. Hartmann (2004) proposes a general model for various scheduling problems (including straddle carriers, AGVs, stacking cranes, and workers that handle reefer containers) that occur in container terminal logistics. The model can be applied to solve several different real world problems for container terminals. The general model is solved by priority rule based heuristics. Kozan and Preston (2006) present an iterative search algorithm that integrates a container

handling schedule with storage strategy in a cyclic mode to determine the optimal storage strategy and container handling schedule. However this study is mainly focused on gate port main. Lee at el. (2009) propose a constructive approach to integrate yard truck scheduling and the storage allocation with the objective of minimizing the weighted sum of the total delay of requests and the total travel time. However they assume that the truck number is very limited and each truck serves exactly one route. Due to the limitations, this approach may not suit real time planning.

For additional references dealing with the management of a whole container terminal, one can refer to Leeper (1988), Hayuth (1994), Mosca et al. (1994), Ramani (1996), Hulten (1997), Merkuryev et al. (1998), Thiers and Janssens (1998), Rizzoli et al. (1999), Veeke and Ottjes (1999), Saanen (2000), Carrascosa et al. (2001), Kim et al. (2002), Meersmans (2002), Veeke and Ottjes (2002), Mattfeld (2003), and Yun and Choi (2003).

From the literature it can be seen that various problems associated with terminal operations have been addressed. Only few studies aim to increase terminal productivity from an integrative view. These studies do not sufficiently address the particular needs of transshipment hubs, but are more on general terminals which emphasize merely on import or export activities. For transshipment hubs, loading and unloading activities need to be considered at the same time, which makes the terminal planning more complex. In this thesis, we study the vehicle dispatching problem by considering loading and unloading containers at the same time.

Also, many of the works on vehicle dispatching problem do not consider the delay at the yard side with the exception of Chen et al. (2007), and they usually assume that YC is always available when needed. This assumption is fine when the yard is not congested, or there are many YCs. However, this might not be the case especially when there is high traffic volume. In our problem, we consider the delay at the yard side, and also include both the loading and discharging jobs simultaneously which are commonly found in the transshipment port. In addition, the PMs are pooled among all the QCs rather than dedicated to a certain QC. We seek to provide an efficient way of dispatching vehicles to minimize the makespan time at the quay side for a given number of container jobs by considering all equipment.

Furthermore, the dispatching and location problem is studied for inbound containers, and a simulation study is conducted on the real time dispatching concept. The details for the integrated dispatching problems and the simulation study on the real time dispatching are discussed in the following chapters.

3 Integrated Vehicle Dispatching Problem for Transshipment Hubs

In this chapter, we address the dispatching problem for vehicles (or prime movers) in a transshipment hub by considering the quay crane and yard crane capacity. The objective of this problem is to minimize the makespan time at the quay side. This issue is particularly important for a port which uses information technology in making real time decisions because the port can exploit information technology to make full use of the data in making good decisions. A mixed integer programming (MIP) model is developed to formulate the problem. As the existing solver cannot solve the MIP model in reasonable time, we develop two heuristics to tackle the problem. The first method is based on the neighborhood search, while the second method is based on genetic algorithm (GA) and minimum cost flow (MCF) network model.

3.1 Model Development

In this section, we assume that several vessels are being loaded and unloaded simultaneously at a given time interval. The yard location for each container is predetermined, and for the PMs, the main operational decisions are to determine the sequence of jobs for the PMs to perform. In practice, most terminal operators simply dedicate a certain number of vehicles to serve a quay crane using a greedy heuristic, such as the nearest task first. While this greedy approach is easy to implement, it might provide

an inferior solution. Therefore, it is important to develop a model which considers all the equipment together. In this section, we develop a MIP model for this vehicle dispatching problem.

3.1.1 Modeling Assumptions

The following assumptions are made:

- Job sequence and job types for each QC are given; Tasks must be carried out by QCs in the exact order which appears in the QC sequence list;
- Yard location of each job is known;
- Traveling times between any two processing locations are also known;
- PMs are shared among all QCs;
- Number of container jobs, number of PMs, number of QCs and YCs are all known;
- PMs can only take one container at a time;
- YC traveling time will be considered in the handling time;
- Traffic congestion of the PMs at the road is not considered.

3.1.2 Notations

The model parameters are as follows:

K: the set of QCs;

M: the set of PMs;

R: the set of YCs;

N_k: the number of jobs in the working list of QC *k*.

L: the set of loading jobs;

D: the set of discharging jobs;

H: the set of all the jobs, $H= L \cup D$;

N: total number of container jobs including loading and unloading. $N= \sum_{k}^{K} N_k = |L \cup D|$.

(i, α): container job index. The job (i, α) refers to the *i*th job in the sequence list of QC α.

(*S*,*D*): Dummy starting job;

(*E*,*D*): Dummy ending job;

J: the job set which contains all the jobs including dummy starting jobs and dummy ending job, $J=\{(S, D), (E, D)\} \cup H$;

J_S: the job set which contains all the jobs including dummy starting job,

$J_S = \{(S, D)\} \cup H$;

J_E: the job set which contains all the jobs including dummy ending job,

$J_E = \{(E, D)\} \cup H$;

J_r: those jobs using yard crane *r*.

C: a huge constant number.

$h_{1(i,\alpha)}$: the QC handling time of job (i, α). (Loading or Discharging)

$h_{2(i,\alpha)}$: the YC handling time of job (i, α). (Loading or Discharging);

$t_{1(i,\alpha)}$: loaded traveling time of job (i, α) from its origin to its predetermined destination.

$t_{2(i,\alpha)(j,\beta)}$: Empty traveling time from destination of job (i, α) to the origin of next job (j, β).

The decision variables are as follows:

$X^{m}_{(i,\alpha)(j,\beta)}$ =1: PM *m* moves to the origin of next job (j,β) after just finishing job (i,α);

Otherwise 0

$Z^{r}_{(i,\alpha)(j,\beta)}$ =1: the job (j,β) is processed by the same YC *r* immediately after job (i,α);

Otherwise 0

$T_{1(i,\alpha)}$: The starting time for processing container job (i,α) by QC α.

$T_{2(i,\alpha)}$: The starting time for processing container job (i,α) by corresponding YC.

As all tasks can either be a discharging or a loading job, we need to analyze the flow time of these two tasks so that they can be formulated in the model.

- Activity flow time for discharging jobs (Figure 3.1)

The following activity flow time describes the time period that one PM needs to finish a discharging job (i,α). The darker area means the possible waiting times for each PM at both the quay side and the yard side.

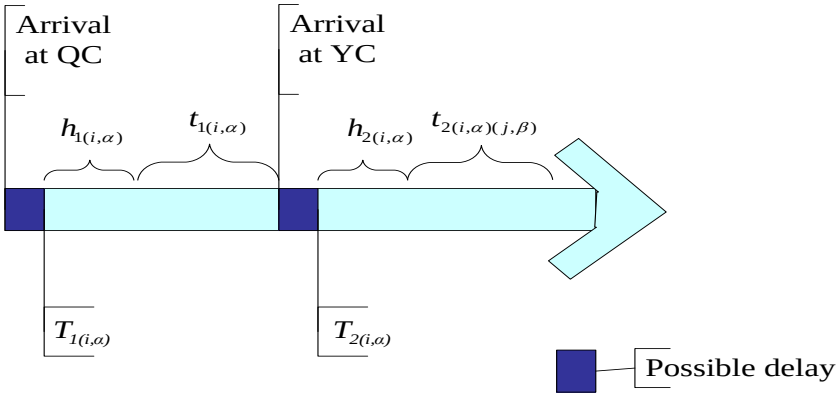

Figure 3.1 Activity flow time for discharging job

For discharging jobs, $T_{1(i,\alpha)}$ is the starting time that QC α releases its i^{th} container job to a PM; $h_{1(i,\alpha)}$ is the handling time for a QC to discharge the container (i,α) to the PM; $t_{1(i,\alpha)}$ is the PM traveling time to move the container (i,α) from the QC to the designated storage location; $T_{2(i,\alpha)}$ is the starting time that the YC picks up the container (i,α) from the PM; $h_{2(i,\alpha)}$ is the handling time for YC to discharge the container (i,α) from the PM to the storage location; and $t_{2(i,\alpha)(j,\beta)}$ is the PM empty traveling time from this YC to the origin of next job (j,β).

- Activity flow time for loading jobs (Figure 3.2)

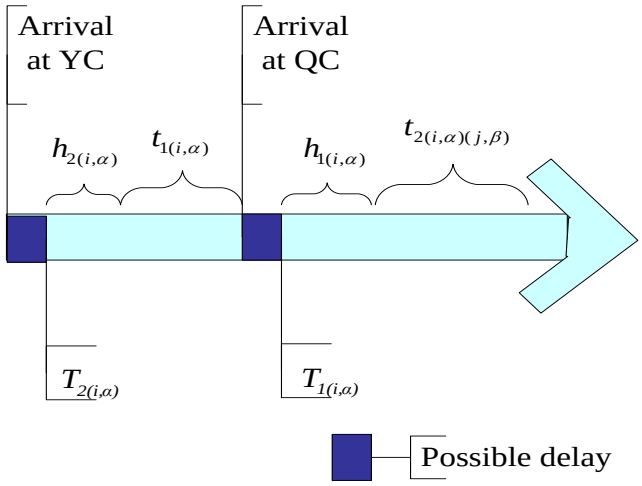

Figure 3.2 Activity flow time for loading job

The following activity flow time describes the time period that one PM needs to finish a loading job (i,α). For the loading job, $T_{1(i,\alpha)}$ and $h_{1(i,\alpha)}$ refer to the starting time and handling time of QC respectively, and $T_{2(i,\alpha)}$ and $h_{2(i,\alpha)}$ refer to the starting time and handling time of YC respectively. A PM retrieves a job (i,α) from its storage location at

the yard side at time $T_{2(i,\alpha)}$ and travels to its destination, which is the QC α. It then goes to next job (j,β) after QC α picks up the container (i,α). $t_{1(i,\alpha)}$ is the PM traveling time from YC to QC for job (i,α) and $t_{2(i,\alpha)(j,\beta)}$ is to the PM empty traveling time from QC α to the origin of next job (j,β).

3.1.3 Model Formulation

Objective:

Min: $Max(T_{1(N_\alpha,\alpha)} + h_{1(N_\alpha,\alpha)})$ (3.1)

Constraint

- **Resource constraints**

$$\sum_{(i,\alpha)\in J_S} \sum_{m=1}^{|M|} X^m_{(i,\alpha)(j,\beta)} = 1, \forall (j,\beta) \in H \quad (3.2)$$

$$\sum_{(j,\beta)\in J_E} \sum_{m=1}^{|M|} X^m_{(i,\alpha)(j,\beta)} = 1, \forall (i,\alpha) \in H \quad (3.3)$$

$$\sum_{(i,\alpha)\in J_S} X^m_{(i,\alpha)(l,\gamma)} = \sum_{(j,\beta)\in J_E} X^m_{(l,\gamma)(j,\beta)}, \forall (l,\gamma) \in H, m \in M \quad (3.4)$$

$$\sum_{(j,\beta)\in H} X^m_{(S,D)(j,\beta)} = 1, \ \forall \ m \in M. \quad (3.5)$$

$$\sum_{(i,\alpha)\in H} X^m_{(i,\alpha)(E,D)} = 1, \ \forall \ m \in M. \quad (3.6)$$

$$\sum_{(i,\alpha)\in J_r \cup (S,D)} Z^r_{(i,\alpha)(j,\beta)} = 1, \forall (j,\beta) \in J_r, \forall r \in R \quad (3.7)$$

$$\sum_{(j,\beta)\in J_r \cup (E,D)} Z^r_{(i,\alpha)(j,\beta)} = 1, \forall (i,\alpha) \in J_r, \forall r \in R \quad (3.8)$$

$$\sum_{(j,\beta)\in J_r} Z^r_{(S,D)(j,\beta)} = 1, \ \forall \ r \in R. \quad (3.9)$$

$$\sum_{(i,\alpha)\in J_r} Z^r_{(i,\alpha)(E,D)} = 1, \ \forall \ r \in R. \tag{3.10}$$

- **Time constraints for a given job**

$$T_{1(i,\alpha)} + h_{1(i,\alpha)} + t_{1(i,\alpha)} \le T_{2(i,\alpha)}, \forall (i,\alpha) \in D. \tag{3.11}$$

$$T_{2(i,\alpha)} + h_{2(i,\alpha)} + t_{1(i,\alpha)} \le T_{1(i,\alpha)}, \forall (i,\alpha) \in L \tag{3.12}$$

- **Sequence dependent times for different resources**

QC: Two jobs served by the same QC must be set apart at least a certain handling time.

$$T_{1(i+1,\alpha)} - T_{1(i,\alpha)} \ge h_{1(i,\alpha),} \forall (i+1,\alpha), (i,\alpha) \in H \text{ , } i = 1, 2, \ldots, N_\alpha - 1, \ \alpha \in K. \tag{3.13}$$

PM: Two jobs served by the same PM must be set apart at least a certain time.

$$T_{1(j,\beta)} - (T_{2(i,\alpha)} + h_{2(i,\alpha)} + t_{2(i,\alpha)(j,\beta)}) \ge C(X^m_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha), (j,\beta) \in D, \ m \in M \tag{3.14}$$

$$T_{2(j,\beta)} - (T_{1(i,\alpha)} + h_{1(i,\alpha)} + t_{2(i,\alpha)(j,\beta)}) \ge C(X^m_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha), (j,\beta) \in L, \ m \in M \tag{3.15}$$

$$T_{1(j,\beta)} - (T_{1(i,\alpha)} + h_{1(i,\alpha)} + t_{2(i,\alpha)(j,\beta)}) \ge C(X^m_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha) \in L, \ (j,\beta) \in D, \ m \in M \tag{3.16}$$

$$T_{2(j,\beta)} - (T_{2(i,\alpha)} + h_{2(i,\alpha)} + t_{2(i,\alpha)(j,\beta)}) \ge C(X^m_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha) \in D, \ (j,\beta) \in L, \ m \in M \tag{3.17}$$

YC: Two jobs served by the same YC must be set apart at least a certain handling time.

$$T_{2(j,\beta)} - T_{2(i,\alpha)} - h_{2(i,\alpha)} \ge C(Z^r_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha), (j,\beta) \in H, r \in R \tag{3.18}$$

- $X^m_{(i,\alpha)(j,\beta)}$=0 OR 1; $\forall (i,\alpha), (j,\beta) \in J, m \in M.$ (3.19)
- $Z^r_{(i,\alpha)(j,\beta)}$=0 OR 1; $\forall (i,\alpha), (j,\beta) \in J, r \in R.$ (3.20)
- $T_{1(i,\alpha)}, \ T_{2(i,\alpha)} \ge 0, \ \forall (i,\alpha) \in H, \ i = 1, 2, \ldots, N_\alpha, \alpha \in K.$ (3.21)

In the objective function (3.1), the makespan of finishing a given set of jobs at the quay side based on the current equipment configuration is minimized.

Constraints (3.2) to (3.10) are resource constraints. Among these constraints, (3.2) to (3.6) are for quay side, while the rest are for yard side. (3.2) and (3.3) imply that every container job in *H* has one predecessor and one successor and served by exactly one PM. Constraint (3.4) ensures the continuity of each PM route. Constraints (3.5) and (3.6) are for dummy starting node and dummy ending node on PM sequence. The resource constraints for yard crane (3.7) to (3.10) are almost the same as those for quay side.

Constraints (3.11) to (3.12) are for time constraints which force the starting time at the QC and YC for every job must be set apart at least by the traveling time between QC and YC and the handling time.

Constraints (3.13) to (3.18) are for the sequence dependent time constraints for different resources which are similar with the parallel machine scheduling problem with precedence constraints and multiple traveling salesmen problem with precedence constraints. They imply that two jobs served by the same QC/PM/YC must be set apart at least a certain processing time. Constraint (3.13) means two jobs served consecutively by the same QC must be set apart at least the handling time of QC. Similarly, constraint (3.18) means two jobs served consecutively by the same YC have to be set apart at least the handling time of YC. Constraints (3.14-3.17) refer to the sequence dependent time constraints for the PM depending on the types of the job pairs such as discharge-discharge, load-load, load-discharge and discharge-load. Constraints (3.19), (3.20) and (3.21) are non-negative and integer restrictions.

3.2 Proposed Heuristic Methods

The dispatching model presented in Section 3.1 will determine both the optimal PM and YC work sequences. We run a numerical experiment consisting of 2 QCs, 3YCs and 2PMs to test the efficiency of this model. The model is solved with CPLEX 11.0 and implemented by C++ programming performed on a 2.4 GHz PC with 2 GB RAM. We find that the computation time for solving this MIP model becomes very large when the number of jobs increases from 6 to 16(shown in Figure 3.3).

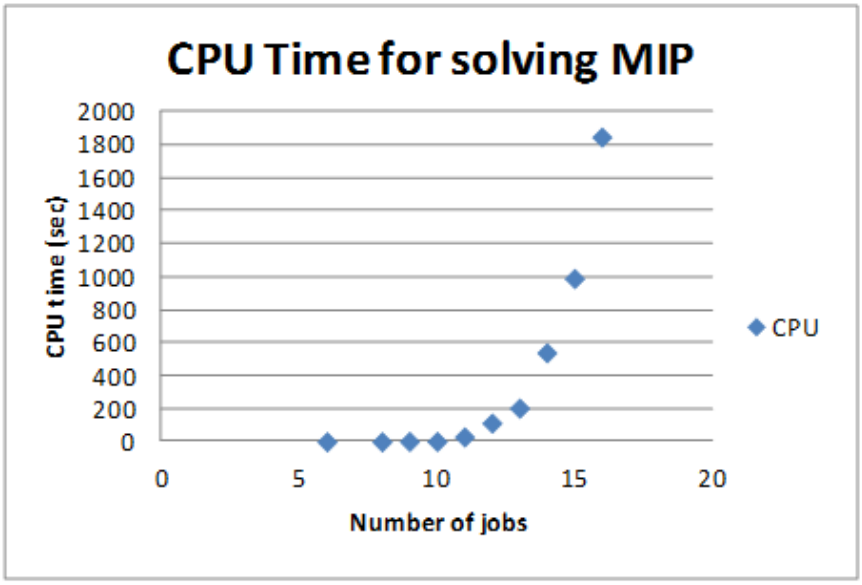

Figure 3.3 CPU time for solving MIP model

In view of the long solving time even for this small size problem, it is not practical to apply this MIP model directly for a real-scale terminal operation. Hence we propose a solution framework as shown in Figure 3.4 to address this issue. The main idea of this framework is to search for PM sequences, and given these PM sequences, the evaluation

model will determine the YC sequences. Eventually, we hope this framework will be able to find good PM sequences.

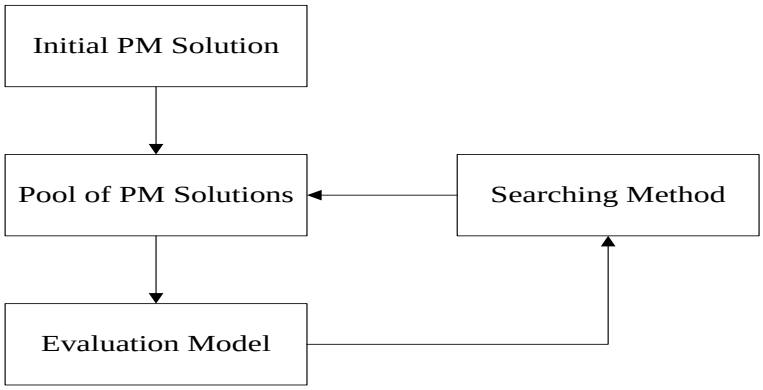

Figure 3.4 A framework of proposed heuristics

We now discuss the concepts of this framework. In this framework, we first generate an initial PM sequence. This sequence can either be randomly generated or based on some heuristic methods such as the minimum cost flow (MCF) network model. Based on this sequence, we create a new set of sequences. The performances of these sequences will be computed based on an evaluation model by considering different YC sequencing rules. To improve the quality of the solutions, search methods will be employed to generate a new pool of sequences. This process repeats until some stopping criteria are met.

For the evaluation module, we need to compute the makespan of the quay side activities for the given PM sequence by considering the delay at the quay side as well as the yard side. We propose two methods. The first one is based on the reduced MIP model and the second one is a simulation approach based on First Come First Serve (FCFS) YC sequencing rule. The reduced MIP model is essentially the same model that we have

presented in Section 3.1, except that the PM job sequence is given. This means that all the values of $X^{m}_{(i,\alpha)(j,\beta)}$ are fixed and the reduced MIP model for the small size problem can be solved effectively since the size of the MIP model is reduced significantly. However, we expect the computational performance to deteriorate when the number of jobs becomes large because it is still a MIP model. For the FCFS-based simulation model, we assume that the YC will serve the jobs by using the FCFS rule according to the PM arrival time at the YC. Since the PM sequence is given and by applying the FCFS rule, we can determine the time the PM completes its activity at the yard side and quay side using discrete event scheduling approach. As we do not need to solve any optimization models, the computation time is very fast compared with the reduced MIP model. In the numerical experiment, we will show that the performance for the simulation approach is quite close to the reduced MIP model, and so we will adopt the simulation approach for large size problems due to its computation efficiency.

For the searching method, we adopt two different methods to generate new PM job sequences: one is the Variable Neighborhood Search (VNS) method; and the other is the GA-MCF approach. In the following subsections, we will discuss the MCF, VNS and GA-MCF methods in more detail.

3.2.1 Minimum Cost Flow (MCF) Model

The goal of this minimum cost flow (MCF) model is to find a schedule that will minimize the impact of delays and maximize the utilization of the vehicles (Cheng 2005). However in our work, we intend to use the MCF as a mean to generate good PM sequences. For a

more detailed review of the minimum cost flow network models, one can refer to Vis et al. (2001) and Vis et al. (2005), Potvin et al. (1992) and (1993).

In the MCF model that we use in our work, we assume the ready times for the jobs are given and they are independent of the PM sequence. Moreover, it also assumes that the YC is always available which is to say that the waiting time is not considered at yard side. The model then seeks to determine the PM sequence which has the least deviations from the ready times of the tasks.

Note that these ready times used in our MCF model are not the actual ready times in the operations, because we do not consider the delay, the interaction between equipment and the job sequences. In other words, these ready times can be viewed as the artificial ready times, and we fix these times to help us to find the PM sequence.

For the subsequent discussion, we will use indices *i* and *j* to denote the container jobs, and *S* and *E* to denote the dummy starting and ending jobs respectively. Our model can also be viewed as a directed graph G (*J*, *A*) where *J* denotes the set of nodes and *A* denotes the set of arcs. All container jobs in the set *H* (contains both discharging and loading jobs) and the dummy starting and ending jobs are represented as nodes in *G*. If two jobs are served by the same vehicle, there is a directed arc connecting them. We need to determine *m* routes from node *S* to node *E* when there are *m* vehicles deployed to serve jobs in the set *H*. The cost of the arc is represented by the deviation of the ready times and our objective is to minimize the overall network cost.

Let X_{ij} be the decision variable and it is assigned 1 when job j is performed immediately after job i by the same vehicle and C_{ij} be the cost parameter representing the deviation of the ready time for job j after the vehicle completes job i and begins to process job j.

The model can be formulated as follows.

MCF Model

Minimize $$\sum_{i\in J}\sum_{j\in J} C_{ij}X_{ij} \tag{3.22}$$

Subject to:

$$\sum_{i\in H} X_{Si} = m \tag{3.23}$$

$$\sum_{i\in H} X_{ij} = 1, \text{ for } j \in H \tag{3.24}$$

$$\sum_{i\in H} X_{ji} = 1, \text{ for } j \in H \tag{3.25}$$

$$\sum_{i\in H} X_{iE} = m \tag{3.26}$$

$$X_{ij} \in [0,1], \text{ for } i,j \in J \tag{3.27}$$

Equation (3.22) states the objective which minimizes the total cost of the flow. Constraints (3.23), (3.24), (3.25) and (3.26) are the flow conservation equations for the m vehicles. Constraints (3.27) limit the flow to not more than 1.

This problem can be solved efficiently and it can guarantee the solution obtained to be binary. In the following, we will discuss on how to compute C_{ij}.

The arcs from the dummy starting node *S* to all container job nodes, and from all container job nodes to the dummy ending node *E* are assigned with zero cost, i.e., C_{Si}=0 and C_{iE}=0 for all *i*.

To compute the cost of arcs between two container job nodes, we need to know the ready times of these two jobs at their respective QCs, as well as the traveling time and the handling times.

Let t_i be the ready time for the QC to pick up (for discharging) or drop off (for loading) containers for job *i*. Let t_{ij} denotes the time interval between the time when the PM starts to do the job at QC for job *i* and the time that it is ready to perform job *j* at the QC location for job *j*. Hence, $t_i + t_{ij}$ is the time that the PM arrives at the QC which is assigned to process job *j*, and C_{ij}, which measures the deviation of the ready time for job *j* after serving job *i* is given as follows.

$$C_{ij} = \begin{cases} t_i + t_{ij} - t_j & \text{if } t_i + t_{ij} - t_j \geq 0 \\ \gamma(t_j - t_i - t_{ij}) & \text{otherwise} \end{cases}$$

where γ>0 and it is a constant.

Noted that γ is a parameter that gives the relative weight between being early and being late. Being late would cause the QC to wait, while being early, will not only cause the PM

to wait, but also may result in infeasibility due to the fact that the QC sequence is violated. We have tuned the parameter γ during the numerical runs.

The computation of t_{ij} depends on the types of the job pair (i,j) such as load-load, load-discharge, discharge-load and discharge-discharge (see Figure 3.5). For example, a discharge-load pair involves the handling time at the QC for job i (h_{1i}), the traveling time from the QC to YC for job i (t_{1i}), the handling time at the YC for job i (h_{2i}), the traveling time between the destination location (YC) of job i to the origin location (YC) of job j (t_{2ij}), the handling times at the YC for job j (h_{2j}), and the travelling time from YC to QC for job j (t_{1j}).

Hence for $i \in D$, and $j \in L$

$$t_{ij} = h_{1i} + t_{1i} + h_{2i} + t_{2ij} + h_{2j} + t_{1j}$$

Similarly, we can define t_{ij} for other cases

$$t_{ij} = h_{1i} + t_{1i} + h_{2i} + t_{2ij} \qquad \text{for } i \in D, \text{ and } j \in D$$

$$t_{ij} = h_{1i} + t_{2ij} + h_{2j} + t_{1j} \qquad \text{for } i \in L, \text{ and } j \in L$$

$$t_{ij} = h_{1i} + t_{2ij} \qquad \text{for } i \in L, \text{ and } j \in D$$

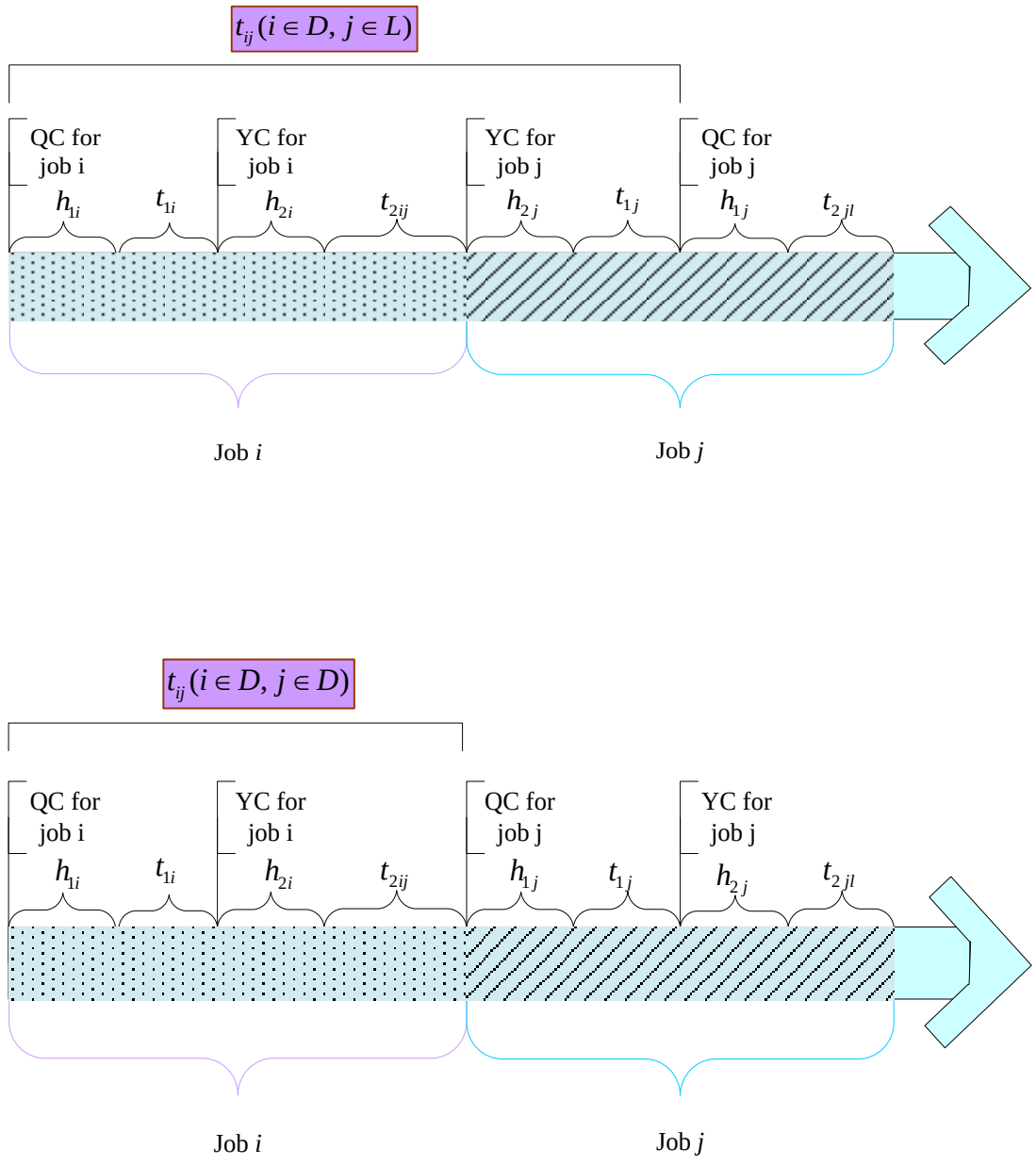

Figure 3.5 Four cases of t_{ij} composition (1)

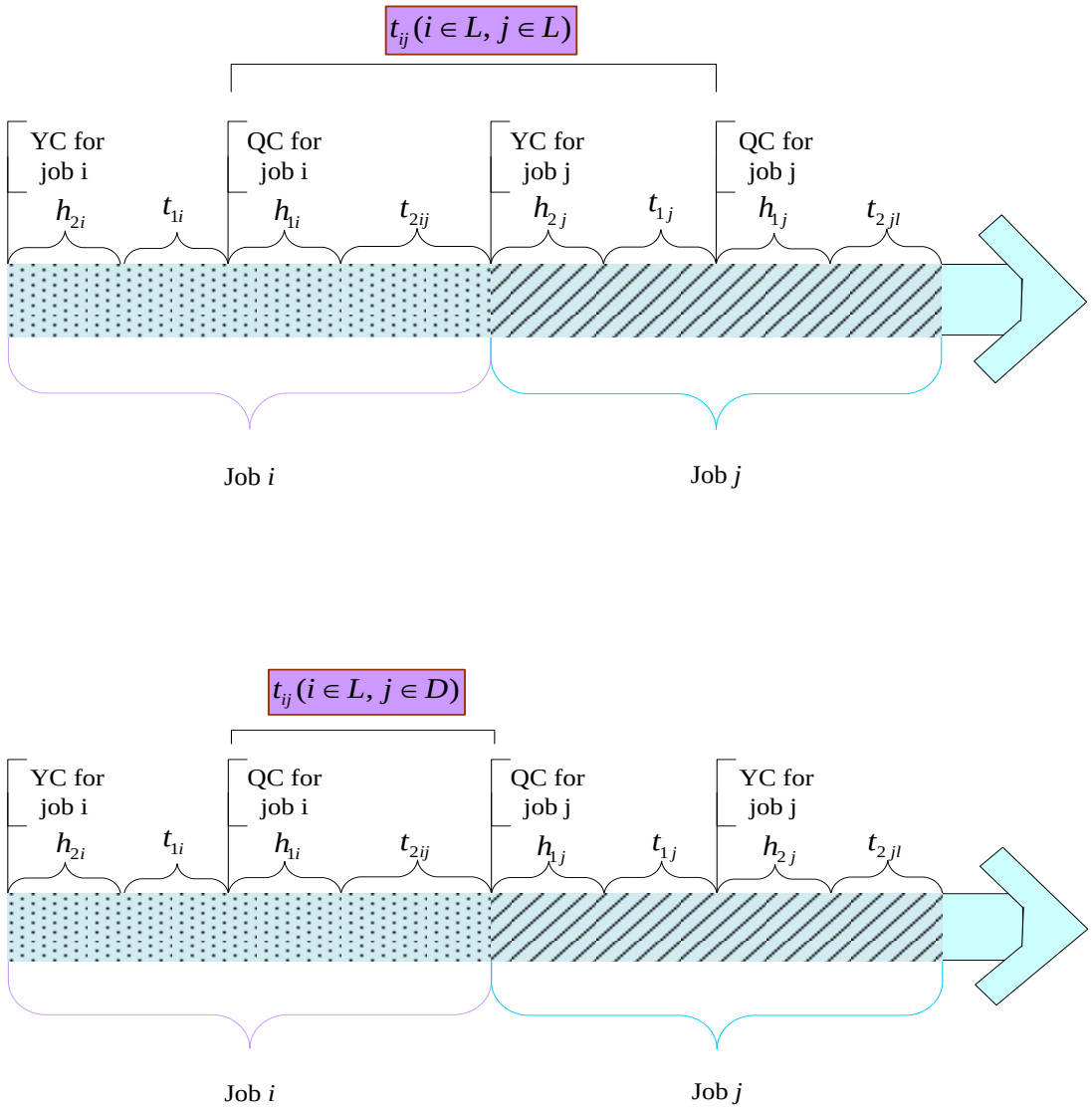

Figure 3.6 Four cases of t_{ij} composition (2)

The merit of this model is that it can solve the vehicles deployment problem efficiently when the ready times t_i for all the jobs, and the times between jobs t_{ij} are given. However the drawback is that it assumes YC is always available, which is to say that the waiting time is not considered at yard side, and moreover by fixing the ready time, we indirectly

assume that the PM can always start the job at the QC at the given ready time, and the actual waiting times for the QC and PM are ignored.

Choosing the "right" ready times is important because some ready times may give infeasible PM sequences while some ready times may give good or even optimal PM sequences. In theorem 1, we show that there exists a set of ready times which will give the optimal PM sequence.

Let v^* be the optimal PM sequences that minimize the makespan at the quay side, and MCF(***t***) be the optimal PM sequences obtained by solving the MCF model given the ready times ***t***., where ***t*** is a vector of the ready times for all jobs.

Theorem 1

There exists an optimal vector of ready times ***t**** for all jobs such that v^*= MCF(***t****).

Proof: Given an optimal PM sequence in v^*, we first fix the ready time for the first job to be zero. We then sequentially set the ready times for all the subsequent jobs by using the following formula. The ready time for job *j*, $t_j^* = t_i^* + t_{ij}$ when there is a PM serving job *j* following job *i* at the optimal PM sequence and the ready time for job *i* is t_i^*. Setting the ready times in this way will result in C_{ij} =0 for all the job pairs following the optimal sequence, v^*. Hence the optimal PM sequences, v^*, will give a zero objective function value for the MCF model when the ready times are set at ***t****.

Theorem 1 shows us that we will not lose the optimality when we search on the design space of the ready times. The MCF model is used in two different ways in our proposed heuristic. Firstly, it is used to generate the initial PM sequence given initial ready times for all the jobs. Secondly, it is used as a decoder for the GA approach when the chromosome is represented by ready times.

3.2.2 VNS Based Heuristics

Variable neighborhood search (VNS) is a heuristic used for solving combinatorial and global optimization problems (Mladenovic & Hansen, 1997). It is a simple and effective search procedure that proceeds to a systematic change of neighborhood. The basic procedure is shown in Garcia (2002). Firstly, an initial solution is found. Then there is a two-nested loop in which the core one alters and explores via two main functions so-called 'shake' and 'local search'. The outer loop works as a refresher reiterating the inner loop, while the inner loop carries out the major local search. The procedure tries to find an improved solution within this neighborhood and iterates as long as it keeps improving the solutions until the stopping criterion has been met, while the shake function diversifies the solution.

In this heuristic, the VNS is used to update our initial solution obtained from the MCF method. The updating procedure is repeated until the stopping criterion has been met. The pseudo code for our VNS is given in Figure 3.6 to illustrate the VNS solution scheme for our integrated dispatching problem.

```
Procedure VNS
First obtain the initial solution X
    For P=0, P < P_limit do
                                Repeat
        X' = Shake (P, X);
        X'' = Local Optimisation ( X')
        If X'' is better than X with respect to f(X) then
          X= X'', P=0;
          else P = P+1;
          end;
        Restart the loop;
     end;
     Return X;
end;
```

Figure 3.7 VNS solution scheme

The initial solution, *X*, is obtained by solving the MCF model and we set P =0. The procedure generates a random solution from a neighborhood of *X* of size *P* by shaking, finds a new solution from this random solution by some local search techniques. In order to intensify the search at the neighborhood of this random solution, the local search is repeated k times. The new solution is adopted if it is better than the current solution, and the size of neighborhood, P, is reset to 0. If the current solution is not improved in any of these k attempts, the neighborhood size P is increased by 1. The algorithm stops when P exceeds the maximum limitation, P_limit.

The local optimization is based on exchanging segments between two PM job sequences (shown in Figure 3.7). The procedure of our VNS based heuristics is discussed below.

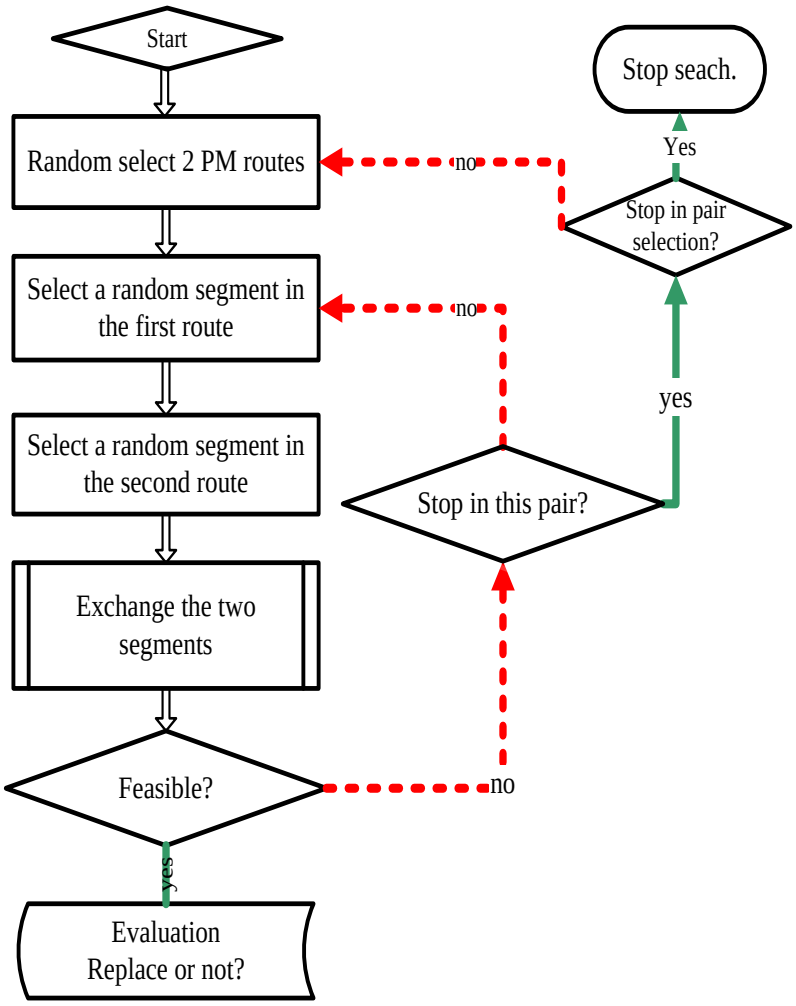

Figure 3.8 The local search in VNS method procedure

Procedure VNS

1) ***Initialization***: Find the initial solution X and set P=0.

2) ***Shake procedure***: Randomly choose a solution X' from the neighborhood of X of size P.

3) ***Local search***: Randomly select two PM sequences from the solution X'.

   a) **a segment of random cardinality in consecutive order in the first sequence is chosen randomly;**

The segment length can be 1, 2, or 3 randomly in consecutive order. The reason behind this is that we do not want to destroy the original sequence and so we keep the cardinality of the segment to be not more than 3.

b) **a segment of random cardinality in consecutive order in the second sequence is chosen randomly;**

c) **exchange these two selected segments**. (As shown in Figure 3.8).

The segment removed from one PM sequence is always inserted in the position within the other PM sequence from where the other segment is removed.

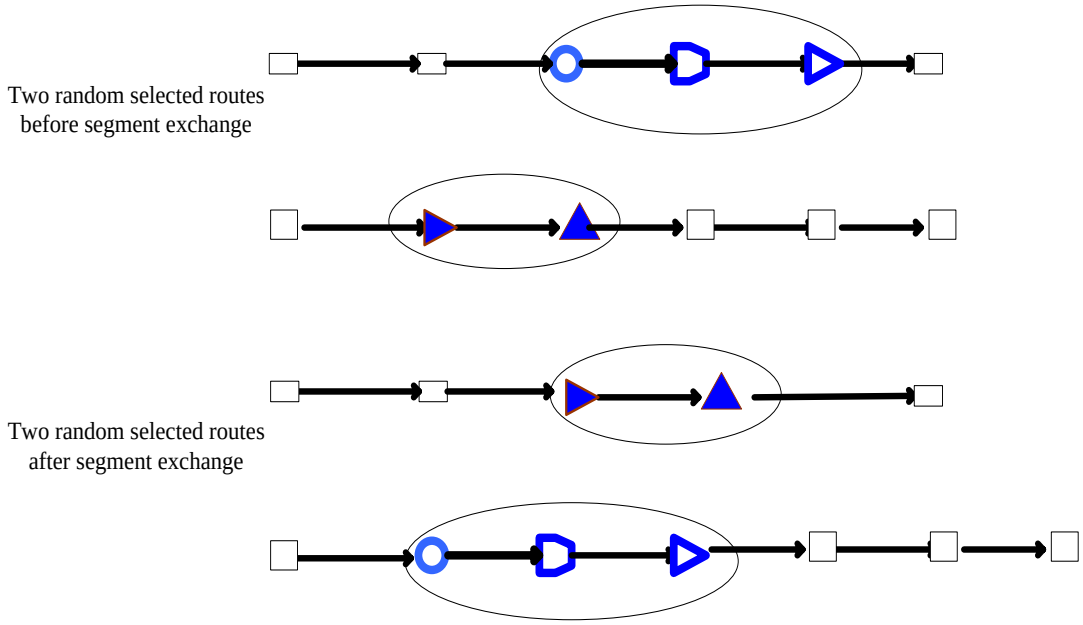

Figure 3.9 Two segments exchange scheme

d) **Feasibility checking**: Check whether the new sequences are feasible based on the given QC sequences.

After exchanging, we should check the feasibility of both of the two PM sequences. Take two jobs *i* and *j* for example; Assume jobs *i* and *j* are specified in the same QC work list and this QC must process job *i* first before it can process job *j* as predefined in the work sequence, therefore job *j* cannot appear in front of job *i* on either of the two PM sequences after exchanging. Otherwise this exchange is infeasible.

If the two new PM sequences are feasible, we can go to the evaluation function in Step 4; Otherwise, return to step b to select another segment in each PM sequence for at most *k* times. To avoid unnecessary search, if a feasible solution cannot be found in *k* times, go to step 2 to shake and select another pair of PM sequences.

4) ***Evaluation:*** Evaluate this new solution X'' in the reduced MIP model or FCFS based simulation function to obtain objective value f(X''). If it is better, accept the new solution X'' and set P=0, and X= X''; otherwise, keep the current solution X. Return to Step 3 and search for *k* times. After the *k* times of local search, let P=P+1, and return to Step 2 to select another solution X' until the P_limit criterion has been met.

3.2.3 GA-MCF Approach

GA is a well-known meta-heuristic approach inspired by the natural evolution of the living organisms. GA works on a population of the solutions simultaneously. It combines the concept of survival of the fittest with structured but randomized information exchange to form robust exploration and exploitation of the solution space. The reason why we choose GA here is that: firstly GA is a well-known meta-heuristic with its efficiency being verified for many problems; Secondly, compared with the VNS heuristic, we need a population-based approach such as GA for better exploration of the solution space. The

procedure of the proposed GA is shown in Figure 3.9. Compared with other traditional GA implementations, we use the ready times to represent the chromosome, and then solve the MCF model to obtain the PM sequences.

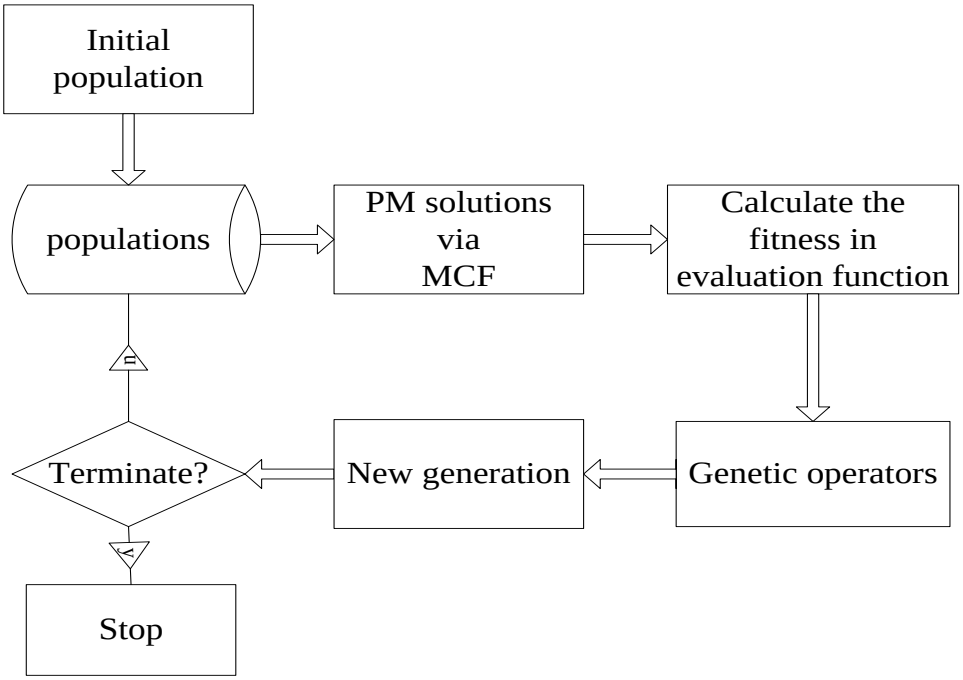

Figure 3.10 The procedure of proposed GA

1) Chromosome representation

We use the ready time to represent the chromosome. The benefit of using this representation compared to using jobs sequence representation is that the neighborhood structure can be preserved more easily when going through the crossover operations. Moreover, the QC sequences can be easily observed. In addition, we have shown in Theorem 1 that by carefully selecting the ready times, we can obtain the optimal PM sequence.

Instead of representing the ready times directly, we represent them using the difference between the ready times for jobs for a given QC sequence. Let t_i be the ready time for job i at a QC, and t_{i+1} be the ready time for the next immediate job at that QC. Then $t_{i+1} = t_i + \Delta_i$ and the chromosome is represented in terms of Δ_i. By representing the chromosome in this way, we can ensure that the ready times for the jobs observe the QC sequence. The GA will search on the Δ_i, and we can easily compute back the ready times given Δ_i.

2) Parents selection

The parent selection strategy describes that how to choose the chromosomes in the current population that will create offspring for the next generation. Generally, it is better that the best solutions in the current generation have more chance to be selected as parents for creating offspring. Hence we propose the use of binary tournament selection. The binary tournament selection will randomly choose two individuals and selects the winner as one parent. For crossover operation, this will be repeated again to select another parent. In order to ensure that the best individual always survives to the next generation, the elitism strategy is used where the best solution is always kept in the population.

3) Crossover

We propose the use of the arithmetic crossover operator to explore the solution space. The new offspring is produced as a linear combination between the parents.

Offspring=λ*Parent1 + (1-λ)*Parent2

where λ is a random number between 0.5 and 1.

4) Mutation

The main task of the mutation operator is to maintain the diversity of the population in the successive generations and to exploit the solution space. For each individual, we first select a random value between 0 and 1 and compare this with the mutation probability *Pm*. Since mutation occurs very infrequently, this is usually set very low (typically *Pm* = 0.001). If the value is less than *Pm* (which is very rare) we perform a mutation operation on the individual. The mutation step is carried out by swapping the value for two randomly chosen genes.

5) Offspring selection

We use a semi-greedy strategy to accept the offspring generated by the genetic operators. In this strategy, an offspring is accepted for the new generation if its fitness is less than the average fitness of its parent(s).

6) Stopping criterion

In order to decrease the computation time, we use two criteria as stopping rules: (1) reaching the maximum number of generations or (2) the standard deviation of the fitness value of chromosomes in the current generation (Tavakkoli-Moghaddam & Safaei, 2006) is below some small value.

3.3 Experiments

We have conducted experiments to assess the solution quality and efficiency of the proposed algorithms. All experiments were performed on a 2.4 GHz PC with 2 GB RAM.

Our heuristics are implemented using C++ and for solving the MIP formulations, the LP solver CPLEX 11.0 by ILOG is used. Each example is solved by VNS and GA 10 times and the means of makespan values (objective value) are reported. Two series of experiments are conducted to evaluate the performance of the proposed heuristics. The first one is a small sized problem. The purpose of the experiments is to compare the difference between the two evaluation functions: the reduced MIP and FCFS based simulation. We find that there is no notable difference between these two evaluation functions. Hence, for a large size problem, we will only use FCFS based simulation to evaluate the performances. In practice, FCFS rule is easier to implement in a container terminal.

In the large size problems, we will compare the performances between the VNS and GA-MCF methods. Two different settings will be used. One has few QCs, YCs and PMs but adopts a longer look ahead strategy (we call it depth-based experiment), while the other one has many QCs, YCs and PMs but adopts a shorter look ahead strategy (we call it breadth-based experiment). The layout parameters and handling time are generated based on a realistic transshipment terminal. The actual quay crane rate is set to 30 boxes per hour and the yard crane rate is set to a uniform distribution of (2, 3) minutes. The predetermined yard location for each container job is randomly assigned.

3.3.1 Small Size Problem

Table 3.1 shows the numerical results of using the reduced MIP model as well as the FCFS simulation model. We can observe that there is not much difference between these

two evaluation methods. However, the reduced MIP model requires more computational time to run.

Table 3.1 Small size comparison results

Case No	Jobs	QC/YC/PM	VNS+Reduced MIP Obj.(min)	Reduced MIP CPU(sec)	VNS+Simulation Obj.(min)	Simu. CPU(sec)
1	11	(2/3/2)	34.79	8.2	34.98	1.66
2	11	(2/3/3)	30.2	5.6	30.2	1.72
3	12	(2/3/2)	40.35	4.66	40.3	1.73
4	13	(2/3/2)	41.2	3.9	40.8	1.78
5	14	(2/3/3)	43.2	2.9	44.1	1.69
6	15	(2/3/2)	48.9	4.53	49.2	2.06
7	16	(2/3/2)	52.62	6.68	52.3	1.9
8	18	(2/3/3)	39.41	18.02	40.01	1.96
9	20	(2/5/3)	43.4	14.3	43.4	2.06
10	25	(2/5/2)	82.6	17.9	81.4	2.01
11	25	(2/5/3)	60.8	27.8	61.8	2.11
12	28	(2/5/3)	63.3	25	63.41	2.21

3.3.2 Large Size Problem

For the large size problem, we only use the FCFS simulation method to evaluate the performance for the solutions obtained by the MCF (initial solutions), VNS and GA-MCF methods.

In the depth-based experiments, we focus on the 2 QCs scheduling environment in which the number of containers varies from 50 to 200, and the number of PMs is from 4 to 8. The experimental results are summarized in Table 3.2, which includes the quay side makespan time solution obtained from the MCF, VNS and GA-MCF methods.

Table 3.2 Depth-based experiments results

Case No	Jobs	QC	YC	PM	Initial Solutions (MCF) Makespan(min)	VNS Makespan (min)	GA-MCF Makespan (min)
1	50	2	2	4	208.39	199.34	171.74
2	50	2	2	6	187.72	180.79	156.34
3	50	2	2	8	175.79	158.68	139.37
4	50	2	3	4	204.58	200.12	192.49
6	50	2	3	8	164.00	160.24	147.15
7	50	2	4	4	203.72	188.26	173.65
9	50	2	4	8	180.44	158.50	147.36
10	100	2	2	4	459.73	459.73	430.78
11	100	2	2	6	438.63	428.64	394.21
12	100	2	2	8	420.01	408.16	370.80
13	100	2	3	6	415.01	411.37	396.04
14	100	2	4	8	379.26	379.26	342.17
15	200	2	2	4	948.37	948.37	895.64
16	200	2	2	6	929.90	929.90	863.34
17	200	2	2	8	906.29	906.29	838.64

Table 3.2 shows that both GA and VNS improve the initial solutions obtained by MCF for the depth-based experiment setting. However, we also observe that when the problem size becomes larger, the improvement of VNS over initial solution becomes negligible, and in some cases, it is not able to improve the initial solution. Table 3.2 also shows that GA consistently outperforms VNS, and the saving can be as high as 10%.

Table 3.3 Breadth-based experiments results

Jobs	QC	YC	PM	Initial Solutions (MCF) Makespan (min)	VNS Makespan (min)	GA-MCF Makespan (min)
50	10	8	15	45.17	44.67	42.31
			20	53.72	46.34	39.61
			30	46.09	39.61	39.61
60	10	8	15	54.72	54.25	53.04
			20	53.25	50.52	48.61
			30	48.95	48.61	48.61
70	10	8	15	71.72	63.40	59.61
			20	67.34	59.77	57.61
			30	66.42	57.61	57.61
80	10	8	15	82.46	73.93	64.88
			20	79.37	76.47	61.81
			30	76.89	73.89	60.61
90	10	8	15	88.25	82.52	78.25
			20	87.37	84.01	72.61
			30	83.69	78.12	71.31
100	10	8	15	98.44	94.05	86.38
			20	97.72	92.58	84.61
			30	94.09	90.28	81.49

Table 3.3 shows the results for the breadth-based experiments. In this series of experiments, we compare the performance obtained from MCF, VNS and GA in a more realistic environment. From the results, we observe that the trend is similar to the depth-based experiment setting. When increasing the PM numbers, the makespan reduces as expected. Figure 3.10 shows the convergence of the GA for a case of 70 jobs, 10 QCs, 8 YCs and 20 PMs. It shows that GA converges very fast and is able to improve the initial solution. The numerical running times for all the cases can be completed within minutes.

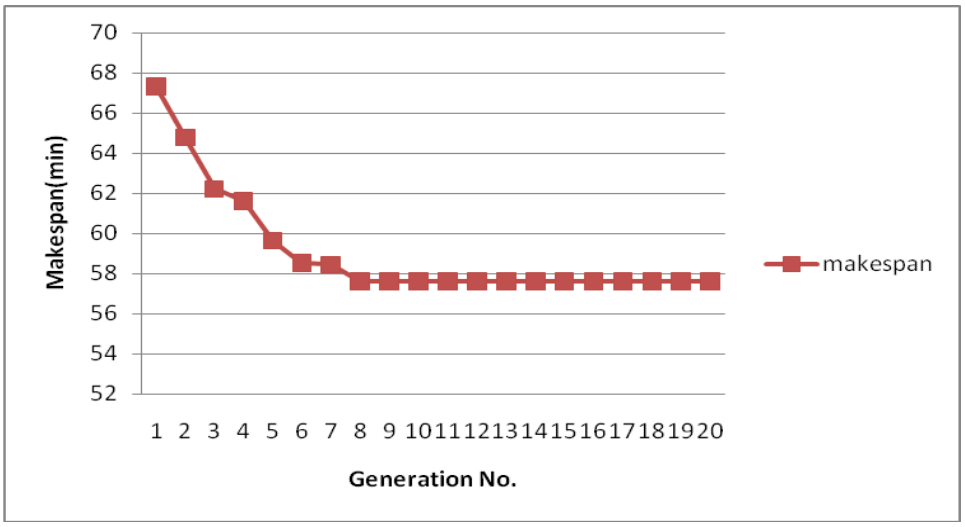

Figure 3.11 Convergence of GA during 20 successive generations for (70 jobs/10 QCs/8 YCs / 20 PMs).

3.4 Summary

In this chapter, we discuss how to assign the container jobs to PMs by considering the capacity of the YC, QC and PM. Two methods are proposed to tackle the problem and they are VNS and GA-MCF methods. We have shown that by using ready times as the chromosome representation, we are able to keep the neighborhood structure and hence good solutions can be found by GA. Moreover, we have shown that by choosing appropriate ready times, we can get the optimal PM sequence. In the numerical runs, we show the superiority of the GA-MCF method over the VNS. In the next chapter, we discuss how to incorporate the storage yard allocation strategy into the decision making so that the overall port performance can be further improved.

4 Container Dispatching and Location Problem

In this chapter, we address the real-time container dispatching and location allocation problem in a container terminal, whose objective is to minimize the makespan time at the quay side. Different from the traditional container dispatching problem, this study aims to solve the vehicle dispatching problem as well as the inbound container allocation problem simultaneously. To tackle this problem, we develop an integrated mathematical model which considers the interaction between all equipment, including YC, QC and vehicles. Due to the exponentially growing search space, it is difficult for existing methods to find a solution in real time, which is important in practical complex working environment. To solve the problem, we propose three heuristic methods which are based on tree search structure; there are: Nested Partition method (NP), Buffered Semi Greedy method (BSG), and Buffered Probabilistic Greedy method (BPG). We also propose a new comparison method to evaluate the performance of these heuristic algorithms, which considers the tradeoff between the elapsed time in exploring the solution space and the quality of the result finally obtained. Extensive experiments are conducted and results show that the proposed heuristic methods can find a promising solution in seconds.

4.1 Model Development

In this chapter, we consider the container dispatching and location problem in both QC and YC sides. The waiting time or delay at both cranes is taken into account in the whole dispatching process. We aim at determining the optimal truck dispatching sequence and

the optimal yard storage location for each discharging container to maximize the productivity in terminals. Before we present the formal problem formulation, we address our model assumptions, notations and constraints first.

4.1.1 Model Assumptions

In our model, we make the following assumptions:

- Job sequence and job types for each QC are given; Tasks must be carried out by QCs in the exact order which appears in the QC sequence list;
- Yard location of each job is assumed to be chosen from several candidate locations;
- Traveling times between any two processing locations are also known;
- PMs are shared among all QCs;
- Number of container jobs, number of PMs, number of QCs and YCs are all known;
- PMs can only take one container at a time;
- YC traveling time will be considered in the handling time;
- Traffic congestion of the PMs at the road is not considered.

4.1.2 Model Formulation

The objective of our model is to minimize the elapsed time to finish all the jobs assigned to the vehicles, which can be formally represented as follows:

Objective:

$$\text{Min: } Max(T_{1(N_\alpha,\alpha)} + h_{1(N_\alpha,\alpha)}) \tag{4.1}$$

We aim at finding an optimal job sequence assignment as well as YC location allocation for each discharging job to minimize the objective value under the following constraints:

- **Resource constraints**

$$\sum_{(i,\alpha)\in J_S}\sum_{m=1}^{|M|} X^m_{(i,\alpha)(j,\beta)} = 1, \forall (j,\beta)\in H \tag{4.2}$$

$$\sum_{(j,\beta)\in J_E}\sum_{m=1}^{|M|} X^m_{(i,\alpha)(j,\beta)} = 1, \forall (i,\alpha)\in H \tag{4.3}$$

$$\sum_{(i,\alpha)\in J_S} X^m_{(i,\alpha)(l,\gamma)} = \sum_{(j,\beta)\in J_E} X^m_{(l,\gamma)(j,\beta)}, \forall (l,\gamma)\in H, m\in M \tag{4.4}$$

$$\sum_{(j,\beta)\in H} X^m_{(S,D)(j,\beta)} = 1,\ \forall\ m\in M. \tag{4.5}$$

$$\sum_{(i,\alpha)\in H} X^m_{(i,\alpha)(E,D)} = 1,\ \forall\ m\in M. \tag{4.6}$$

$$\sum_{(i,\alpha)\in J_r\cup(S,D)} Z^r_{(i,\alpha)(j,\beta)} = 1, \forall (j,\beta)\in J_r, \forall r\in R \tag{4.7}$$

$$\sum_{(j,\beta)\in J_r\cup(E,D)} Z^r_{(i,\alpha)(j,\beta)} = 1, \forall (i,\alpha)\in J_r, \forall r\in R \tag{4.8}$$

$$\sum_{(j,\beta)\in J_r} Z^r_{(S,D)(j,\beta)} = 1,\ \forall\ r\in R. \tag{4.9}$$

$$\sum_{(i,\alpha)\in J_r} Z^r_{(i,\alpha)(E,D)} = 1,\ \forall\ r\in R. \tag{4.10}$$

- **Time constraints for a given job**

$$T_{1(i,\alpha)} + h_{1(i,\alpha)} + t^r_{1(i,\alpha)} W_{(i,\alpha)r} \le T_{2(i,\alpha)}, \forall (i,\alpha)\in D, r\in R. \tag{4.11}$$

$$T_{2(i,\alpha)} + h_{2(i,\alpha)} + t_{1(i,\alpha)} \le T_{1(i,\alpha)}, \forall (i,\alpha)\in L \tag{4.12}$$

- **Sequence dependent times for different resources**

QC: Two jobs served by the same QC must be set apart at least a certain handling time.

$$T_{1(i+1,\alpha)} - T_{1(i,\alpha)} \ge h_{1(i,\alpha),}\forall (i+1,\alpha),(i,\alpha)\in H,\ i = 1, 2, \ldots, N_\alpha - 1,\ \alpha\in K. \tag{4.13}$$

PM: Two jobs served by the same PM must be set apart at least a certain time.

$$T_{1(j,\beta)} - (T_{2(i,\alpha)} + h_{2(i,\alpha)} + t^{r}_{2(i,\alpha)(j,\beta)} W_{(i,\alpha)r}) \geq C(X^{m}_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha),(j,\beta) \in D,\ m \in M,\ r \in R \tag{4.14}$$

$$T_{2(j,\beta)} - (T_{1(i,\alpha)} + h_{1(i,\alpha)} + t_{2(i,\alpha)(j,\beta)}) \geq C(X^{m}_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha),(j,\beta) \in L,\ m \in M \tag{4.15}$$

$$T_{1(j,\beta)} - (T_{1(i,\alpha)} + h_{1(i,\alpha)} + t_{2(i,\alpha)(j,\beta)}) \geq C(X^{m}_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha) \in L,\ (j,\beta) \in D,\ m \in M \tag{4.16}$$

$$T_{2(j,\beta)} - (T_{2(i,\alpha)} + h_{2(i,\alpha)} + t^{r}_{2(i,\alpha)(j,\beta)} W_{(i,\alpha)r}) \geq C(X^{m}_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha) \in D,\ (j,\beta) \in L,\ m \in M, r \in R \tag{4.17}$$

YC: Two jobs served by the same YC must be set apart at least a certain handling time.

$$T_{2(j,\beta)} - T_{2(i,\alpha)} - h_{2(i,\alpha)} \geq C(Z^{r}_{(i,\alpha)(j,\beta)} - 1), \forall (i,\alpha),(j,\beta) \in H, r \in R \tag{4.18}$$

- **Yard Location for Discharging jobs constraints**

$$\sum_{(i,\alpha)\in D} W_{(i,\alpha)r} \leq |D|, \forall r \in R \tag{4.19}$$

$$\sum_{r\in R} W_{(i,\alpha)r} = 1, \forall (i,\alpha) \in D \tag{4.20}$$

$X^{m}_{(i,\alpha)(j,\beta)}$ =0 OR 1; $\forall (i,\alpha), (j,\beta) \in J, m \in M.$ (4.21)

$Z^{r}_{(i,\alpha)(j,\beta)}$ =0 OR 1; $\forall (i,\alpha), (j,\beta) \in J, r \in R.$ (4.22)

$T_{1(i,\alpha)}, T_{2(i,\alpha)} \geq 0, \forall (i,\alpha) \in H, i = 1, 2, \ldots, N_{\alpha}, \alpha \in K.$ (4.23)

$$W_{(i,\alpha)r} \in 0 or 1, \forall (i,\alpha) \in D, r \in R \tag{4.24}$$

Constraints (4.2) to (4.10) are resource constraints. Among these constraints, (4.2) to (4.6) are for quay side, while the rest are for yard side. (4.2) and (4.3) imply that every container job in *H* has one predecessor and one successor and served by exactly one PM. Constraint (4.4) ensures the continuity of each PM route. Constraints (4.5) and (4.6) are

for dummy starting node and dummy ending node on PM sequence. The resource constraints for yard crane (4.7) to (4.10) are almost the same as those for quay side.

Constraints (4.11) to (4.12) are for time constraints which force the starting time at the QC and YC for every job must be set apart at least by the traveling time between QC and YC and the handling time.

Constraints (4.13) to (4.18) are for the sequence dependent time constraints for different resources which are similar with the parallel machine scheduling problem with precedence constraints and multiple traveling salesmen problem with precedence constraints. They imply that two jobs served by the same QC/PM/YC must be set apart at least a certain processing time. Constraint (4.13) means two jobs served consecutively by the same QC must be set apart at least the handling time of QC. Similarly, constraint (4.18) means two jobs served consecutively by the same YC have to be set apart at least the handling time of YC. Constraints (4.14) to (4.17) refer to the sequence dependent time constraints for the PM depending on the types of the job pairs such as discharge-discharge, load-load, load-discharge and discharge-load. Constraints (4.19) and (4.20) are yard location constraints for each discharging job. Constraints (4.21) to (4.24) are non-negative and integer restrictions.

4.2 Solution Scheme

Since the variables grow dramatically with the number of jobs, YC destinations and vehicles, it is rather difficult and time-consuming to solve the mathematical model with

commercial software like CPLEX due to the complexity of the problem. Therefore we use another way to represent the problem which is based on sequence assignment. In our problem, each job represents the allocation of a container to a vehicle, either from QC to YC or in the reverse direction. The YC location in the imported container is unknown beforehand and needs to be determined. Thus, a scheduling algorithm needs to determine the allocating vehicle and YC location for each imported container, as well as job sequences in each vehicle. The goal is to finish all the jobs as soon as possible, i.e., minimize the elapsed time when the last job is finished. Due to the nature of the problem, we find that tree structure is one of the good ways to represent the problem. Formally, we define the problem as follows:

Definition 1: ***Real-Time Container Dispatching and Location Problem***

Given K QCs and R YCs in a terminal, if there are N jobs to be assigned to M vehicles, our goal is to determine the YC destination(if any) and allocating vehicle for each job, and the job sequence in each vehicle, so that the finish time of the last job is minimized.

Figure 4.1 shows a simple example to assign 8 jobs in two QCs to 2 vehicles with 2 YC destinations. Initially, we have sets of jobs in different QCs to get assigned. Each QC is associated with 4 jobs in increasing order of job sequence number. Since it is required that these jobs must be assigned in increasing order of job sequence in the QC, we use a stack to store these jobs on each QC and only the top element is accessible. A scheduling algorithm starts by picking jobs on the top of the QC stacks and assigning them to M_1 or M_2 with YC location in YC_1 or YC_2. For example, if J_{11} is picked and assigned to M_1 with

destination at YC_2, it will be removed from the stack and the scheduler repeats the procedure under the same assignment criteria until finally all the stacks are empty and the 8 jobs are assigned.

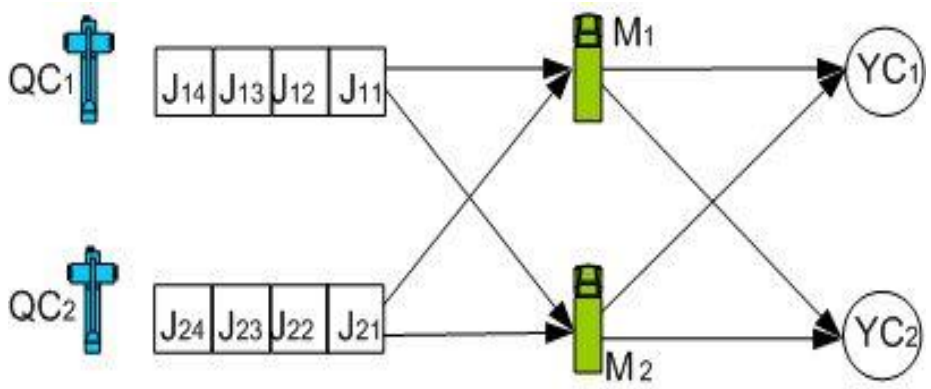

Figure 4.1 A simple example of job assignment

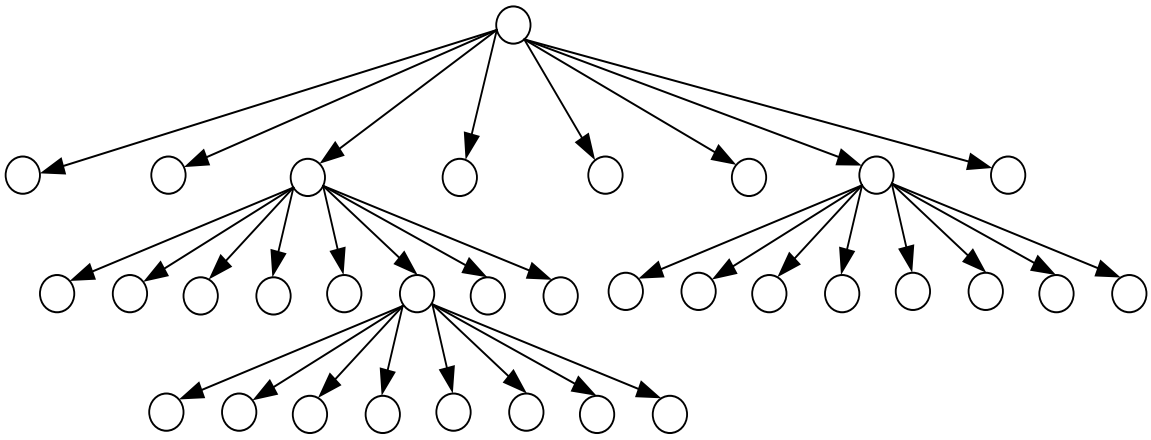

Figure 4.2 Tree representation of search space

We can model the whole scheduling algorithm using a tree structure as shown in Figure 4.2. Initially, no jobs are assigned yet and the tree contains only a root node. Then, in the first step, we have 8 choices of job assignment because we can select either J_{11} from QC_1 or J_{21} from QC_2, assign it to M_1 or M_2 with destination YC_1 or YC_2. It results in 8 subspaces and for each subspace, we can repeat the selection procedure to further partition

the subspace until all the jobs have been assigned. Eventually, we can obtain a tree with height *N*, which is the number of jobs *N*. Each node in the tree represents a job assignment in the scheduling algorithm. Each path from the root to leaf node represents a solution and all the leaf nodes constitute the complete solution space. Thus, we have the following definitions:

Definition 2: ***Partial Solution***

An internal node in the tree structure represents a partial solution, in which not all the jobs have been assigned.

Definition 3: ***Complete Solution***

A leaf node in the tree structure represents a complete solution, in which all the jobs have been scheduled to the vehicles and the YC destination in each job is also determined.

Definition 4: ***Solution Space***

All the complete solutions constitute the solution space.

In this part, we prove that the solution space grows exponentially with the number of jobs. More specifically, we have:

If there are K quay cranes, R yard cranes and N jobs to be assigned to M vehicles, the solution space has a loose lower bound O($\frac{(M \cdot R)^N}{M!}$).

Proof. Given N jobs to be scheduled into a job sequence, we need to determine for each job which YC destination it will be dispatched to and which vehicle will execute the dispatching operation. Thus, each job will have *M*R* choices. Since we have N jobs in a sequence, there will be $(M*R)^N$ instances of job schedules. Among all these schedules, there exist duplicates because we treat all the vehicles equally. For example, assigning a sequence of jobs S_1 to vehicle M_1 and S_2 to vehicle M_2 is identical to assigning S_1 to vehicle M_2 and S_2 to vehicle M_1. Since there are at most $M!$ duplicates for each schedule, the solution space is at least O($\frac{(M \cdot R)^N}{M!}$) when we have not taken into account the detailed job sequence on the vehicle. If that is considered, the solution space is much larger than this lower bound.

The actual solution space is much larger than this loose bound as the job sequences on the vehicles can generate enormous number of cases. A small increase in the number of jobs, YCs and vehicles will cause the solution space to grow exponentially. Thus, an effective operation supporting platform is needed to facilitate real time decision makings of this container dispatching and location problem.

4.3 Proposed Heuristic Methods

Since it is very complex to explore all the tree nodes, based on the tree representation of solution space, we propose three heuristic algorithms: Nested Partition method (NP),

Buffered Semi Greedy method (BSG), and Buffered Probabilistic Greedy method (BPG). The big similarity of these three algorithms is that they all use the tree structure to represent the solution. The difference among them is how to explore the tree structure. The NP method partitions each node into feasible child nodes (subspaces) by determining a job assignment. For each child node, the NP method further explores the remaining jobs by random sampling until it obtains a complete solution. Backtracking is allowed when no feasible candidate exists in the subspaces. The iteration continues until all the jobs have been assigned. Although the NP method is effective in finding a feasible solution, it does not have an upper bound on the performance time. The random sampling and frequent backtracking take too much time when the problem size is large. Thus, we propose two greedy algorithms with a fixed-size buffer, named BSG and BPG respectively. The BSG greedily selects top k partial solutions at each stage, and the next iteration only will expand from these partial solutions. This algorithm works extremely fast when k is small but is likely to miss the global optima. The BPG method improves it by selecting the subspaces in a probabilistic manner. The local top-k optima will be selected with a high probability and the other promising subspaces can also be captured with certain probability.

4.3.1 Nested Partition Method (NP)

Before we present our method based on Nested Partition to solve the container dispatching and location problem, we first provide the necessary preliminary knowledge of Nested Partition.

4.3.1.1 Introduction of Nested Partition (NP)

The Nested Partition method is first proposed by Shi (1997, 2000) for solving global optimization problems. The idea behind this method is to concentrate the computational effort on the most promising solution space that is most likely to contain global optimum. In this method, the solution space is partitioned recursively into subspaces; and random sampling is used to access the potential of each subspace; then the computational effort is focused on the selected most promising subspace. Unlike Semi Greedy method which makes a decision based on local knowledge, the NP method selects a local optima based on global knowledge. In other words, it will foresee the future decisions by random sampling. Thus, it has a higher probability to find a global optimum than Semi Greedy methods.

We consider the following optimization problem:

$$x^* \in \arg\min_{x \in \aleph} f(x)$$

where the solution space $\aleph$ is finite and $f : \aleph \to \Re$ is the performance function to be optimized. In each iteration of the algorithm, we assume that we have a region or sub region of $\aleph$ which is considered to be most promising. Then we partition this most promising region into K sub regions and aggregate the entire surrounding region into one. Next, we calculate the promising index for these K+1 regions by using some random sampling scheme. The most promising region is then selected based on the index values among these K+1 regions. If the surrounding region is selected, the algorithm backtracks to a super region that contains the old most promising region. The algorithm is then partitioned and sampled in the new most promising region in a similar way.

Procedure NP

1) ***Partition***

Partition search region $\Omega(n)$ into $K_{\Omega(n)}$ sub regions: $\Omega_1(n),\ldots,\Omega_{K_{\Omega(n)}}(n)$, and aggregate the surrounding region $\aleph \setminus \Omega(n)$ into one region $\Omega_{K_{\Omega(n)}+1}(n)$.

2) Random Sampling

Randomly sample M_j points from each of the regions $\Omega_j(n)$, j=1, 2,…, $K_{\Omega(n)}$ +1.

x^{j1}, x^{j2},…, x^{jM_j}, j=1, 2,…, $K_{\Omega(n)}$ +1;

and calculate the corresponding performance values,

$f(x^{j1})$, $f(x^{j2})$,…, $f(x^{jM_j})$, j=1, 2,…, $K_{\Omega(n)}$ +1.

3) ***Calculating the Promising Index***

For each region $\Omega_j(n)$, j=1, 2,…, $K_{\Omega(n)}$ +1, calculate the promising index $I(\Omega_j)$

$$\hat{I}(\Omega_j) = \min_{i\in\{1,2,\ldots,M_j\}} f(x^{ji}),\ \text{j=1, 2,…, } K_{\Omega(n)}+1.$$

4) **Backtracking**

Determine the most promising region $\Omega_{\hat{j}_n}$ where

$$\hat{j}_n = \arg\min_{j\in\{1,2,\ldots,M_{\Omega(n)+1}\}} \hat{I}(\Omega_j)$$

If two or more regions are equally promising, break ties arbitrarily. If the promising region refers to a region that is a sub region of $\Omega(n)$, then let this be the most promising region in the next iteration; otherwise, if the promising region refers to a surrounding region, backtrack to the super region of the current most promising region.

4.3.1.2 Apply NP to Our Model

1) Partitioning

The partition must be done in the same manner until the final region which contains only one solution. A better partitioning scheme will lead to an optimal solution in a fast way, while a worse partitioning scheme may backtrack frequently. The NP method tends to perform much better if good solutions are clustered together under the partitioning scheme. To impose such a structure, we introduce the following problem based partitioning scheme: at each stage, we pick a job from the top of the QC stacks and determine its YC location (if necessary) and the vehicle to be assigned to; the next partition is to augment the solution by adding possible combination of job, truck, and YC location. Figure 4.3 shows an example of assigning 4 jobs from 2 QCs to 2 YCs and 2 PMs. In each iteration of the job assignment, there are 8 choices for each job assignment because we can assign any job on the top of the QC's job stack to any vehicle with any YC destination.

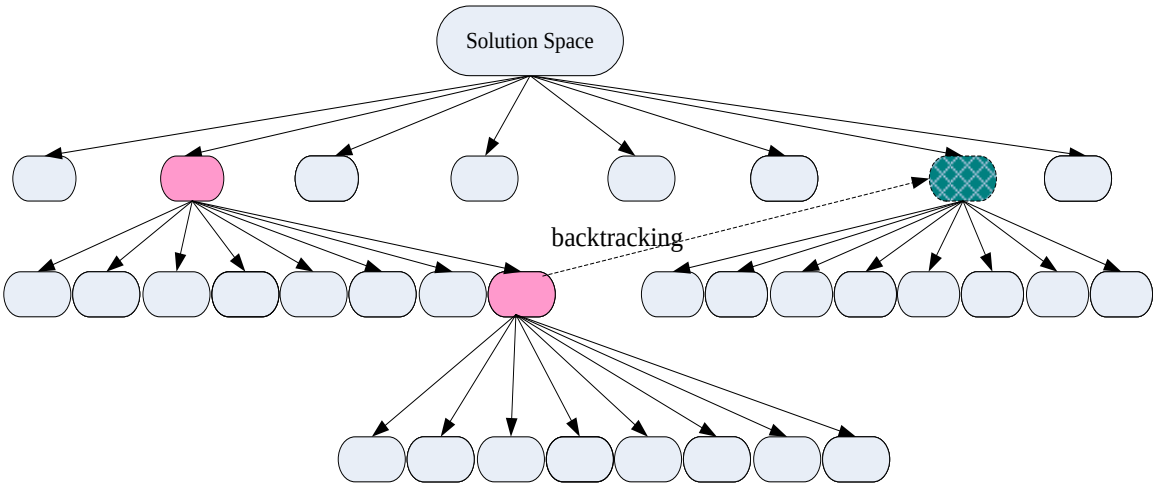

Figure 4.3 An example for Partitioning (2 QCs, 2 PMs, 4 Jobs)

2) Random Sampling

Sampling decides the efficiency of the NP method. As shown in the example of Figure 4.3, in the first iteration, sampling in each region entails finding the remaining job assignments based on the first job assigned, which means finding a completed solution. The method used for random sampling in each region in each stage is not fixed; we can have various random sampling ways. For example, we can use pure random sampling in each region in the above example. However this rough random sampling will result in backtracking through the NP procedure. In this study, we use greedy weighted sampling schemes based on the abovementioned tree based partitioning techniques. Our random sampling incorporates the objective function, that is to say, the better the performance of the assignment, the higher the chance for it to be selected.

For example, we sample the region of the shaded node in Figure 4.3. In the first stage, we select the next assignment based on the performance value of the remaining job assignments. The best objective value has the highest probability to be selected. This procedure mimics the greedy search until all the jobs have been assigned and a complete solution has been obtained, and it still has feasible probability to select any job assignment (the procedures are shown below). This greedy weighted sampling has two main advantages: firstly, it can guarantee the feasibility of the solution; secondly, it incorporates a greedy conception which makes the searching process efficiently.

Weighted Sampling

Assume a region defined by the first k jobs assigned and a predetermined parameter:
Constant $p \in (0,1)$
For *i=k+1: n* DO
generate a random number u uniformly distributed on (0,1)
if $u \leq p$,
let the next assignment be the lowest cost added
else
let the next assignment be a random selection according to a uniform distribution
end
end

The constant parameter *p* can be any number between 0 and 1; when p equals to 1, it is a pure greedy assignment and when p equals to 0, it is a pure uniform assignment. We will test different *p* values in the experiments.

3) Calculating the Promising Index

After using the above random sampling scheme, we need to calculate the promising index for each region in order to pick up the most promising region in the next iteration. For each region $\Omega_j(n)$, j=1, 2,…, $K_{\Omega(n)}$+1, calculate the promising index $I(\Omega_j)$

$$\hat{I}(\Omega_j) = \min_{i \in \{1,2,\ldots,M_j\}} f(x^{ji}), \text{ j=1, 2,…, } K_{\Omega(n)}+1.$$

Here we use the minimum objective value (makespan time) to become the promising index of each region. To get the each objective value, we use simulation to evaluate the performance of such a solution.

4) **Backtracking**

The NP method enables a feasible move from one region to another region or even entire region as long as the promising index indicates that the backtracking is the appropriate move. In our problem, if the index corresponds to the surrounding region, the algorithm backtracks to the super region of the current most promising region. Although the NP method does not require an initial solution, it takes a global view in each stage, and it is flexible to move from one cluster to another cluster based on the index.

4.3.2 Buffered Semi Greedy Method (BSG)

The NP method combines global and local search naturally and demonstrates effective performance in many cases. However, such a heuristic method does not guarantee a lower bound of the solution space. Considerable search regions need to be explored if the backtracking strategy is not well designed. Moreover, to estimate the performance value of a search region, random sampling is adopted to generate a complete solution, which is costly when the number of jobs is large.

We propose a Buffered Semi Greedy (BSG) method which guarantees a maximum search cost and is suitable for the real-time environment. The intuition behind the BSG method is that it extends the nearest-neighbor heuristic to k-nearest-neighbors heuristic. Thus, instead of keeping a record of local optima, we maintain a fixed-size buffer to store k items of local optima in each level of the search space. These k search regions are considered to be more likely to contain the global optima. The search procedure of the BSG Algorithm is shown as follows.

BSG – Buffered Semi Greedy Search Algorithm

1) **while** jobs exist in QC job stacks **do**
2) **for** each job J_i assigned to vehicle M_j with destination YC_r **do**
3) calculate the estimate finish time *f* of current partial solution
4) **if** *f* is smaller than *max(f)* in the buffer, **then**
5) insert current partial solution to buffer
6) **fi**
7) **done**
8) **done**
9) **return** min(f) in the buffer

Figure 4.4 shows an example of the BSG search algorithm to dispatch jobs with working environment in Figure 4.1. We have two QCs, two YCs and two vehicles for job allocation. Initially, we start from the whole space and maintain a buffer with size two to store partial solutions. In the first level, we select two most promising search regions from the eight branches. Note that we determine the performance value based on local knowledge. In other words, we calculate the estimated finish time for the partial solution found so far in each region. The one with the minimum estimated finish time is considered as the most promising. Then, in each iteration downward the tree, 16 search regions will be generated from the two super-regions in the buffer and 2 regions with best performance value are selected and stored in the buffer. The process continues until all the jobs have been assigned.

Such a Semi Greedy search algorithm can work very fast and is guaranteed to finish the job dispatching within a bounded time. If there are *K* QCs, *R* YCs and *M* vehicles, to make a schedule of *N* jobs, at most *k*K*R*M*N* will be explored for a buffer with size *k*. Therefore, the search space is actually linear with respect to the number of QCs, YCs, jobs and vehicles. Besides its efficiency when applied in real-time environment, we observe

that if the job length, i.e., elapsed time to finish a job, is uniformly distributed, BSG can achieve the optimal solution. However, if the jobs are varied in their length, BSG is only able to find local optima and cannot perform well in such a scenario.

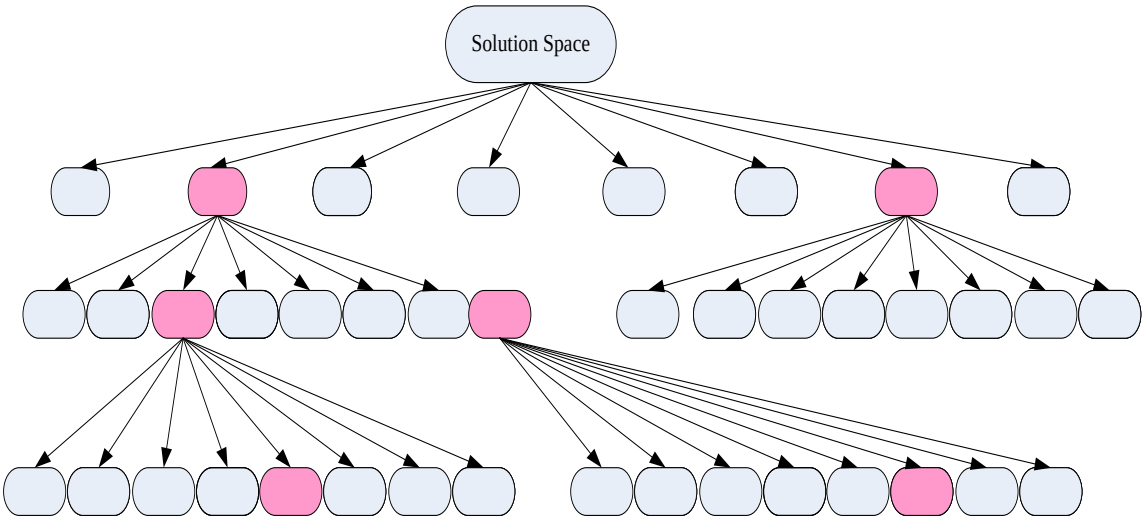

Figure 4.4 Example for BSG method (2 QCs, 2 PMs, 4 Jobs, 2 YC Locations)

4.3.3 Buffered Probabilistic Greedy Method (BPG)

The main disadvantage of BSP is that it only reserves top-k local optima and the constructive mechanism may miss the potential solutions. Since our model is a combinatorial optimization problem, it is likely to have the same performance value in the early stage of search space exploration. In other words, the buffered solutions are not distinguishable enough. Some of the partial solutions will lead to the same complete solution and the potential good solutions for global optima are missed. To capture the global optima with a higher probability, one alternative is to adopt the scheme proposed by Bresina (1996), in which candidates are assigned probabilities of being selected that are

determined by their performance values. In this section, we propose a probabilistic selection scheme which is able to capture potential good solutions outside the top-k local optima.

In this scheme, candidate elements are assigned probabilities of being selected based on their current greedy objective values in every stage until the buffer size is reached. The greedy objective value f here refers to makespan time of this assignment. The bias parameter b_i is introduced here to assign to the i-th ranked element:

$$b_i = \frac{f_{best}}{f_{\mathrm{i}}},$$

where f_{best} is the best objective function value found in current stage, and f_i is the value of the i-th ranked element in this stage. The probability that the i-th ranked candidate element is selected is

$$p(i) = \frac{b_i}{\sum_{j=1}^{|C|} b_j},$$

where set C is the set of all candidate elements in this stage. It contains all the possible assignments extended from previous buffer. Thus, |C|=k*K*M*R. Note that the lower the value of f_i, the larger the value of b_i, and consequently, the higher the value of $p(i)$ making the i-th ranked element more likely to be selected. Similarly to the BSG method, the buffer size k here is fixed, and the selection is ended when the buffer is full. The pseudo code of the algorithm is shown as follows.

BPG – Buffered Probabilistic Greedy Search Algorithm

1) **while** jobs exist in QC job stacks **do**
2) **for** each job J_i assigned to vehicle M_j with destination YC_r **do**
3) calculate the estimated finish time *f* of current partial solution
4) **if** $f_i < f_{best}$, **then** $f_{best} = f_i$
5) **fi**
6) **done**
7) **for** each f_i **do**
8) calculate $b_i = \frac{f_{best}}{f_i}$
9) $sum(b) += b_i$
10) **done**
11) **for** each b_i **do**
12) calculate $p(i) = \frac{b_i}{sum(b)}$
13) **done**
14) **while** buffer is not full **do**
15) pick a random value ρ in [0,1]
16) find *i* such that $\sum_{j=0}^{i-1} p(i) \le \rho < \sum_{j=0}^{i} p(i)$
17) **if** search region *i* is not in the buffer, **then**
18) insert it to buffer
19) **fi**
20) **done**
21) **done**
22) **return** min(f) in the buffer

4.4 Experiments

To assess the solution quality and efficiency of the proposed algorithms, we conduct extensive experiments with different parameter settings. The layout parameters and handling time are generated based on a realistic transshipment terminal. As in practice, three potential candidate yard locations are given to a set of discharging jobs. These

candidate locations are dominated by YC current positions; they can be different yard blocks or different slots in the same block. The YC traveling time along the yard block is considered in the YC handling time of each job. Both the processing times in quay crane and yard crane are set to follow a normal distribution with a variance of 0.2; namely normal distribution (2, 0.2) and (3, 0.2).

In the experiments, we test the robustness of each algorithm by evaluating their performance in terms of different crucial parameters. We also compare these three methods based on a time-quality measure, which is defined as:

$$\delta = \frac{quality}{time},$$

where quality is defined as the ratio of the performance value of the current solution to that of the global optimal solution:

$$quality = \frac{f}{f_{opt}}$$

Since the three algorithms get the same f_{opt} with identical parameter setup of jobs, vehicles, QCs and YCs, we can set the optimal solution as any small positive constant value; and the quality factor is normalized for comparison purpose. The intuition behind the measure is that there exists tradeoff in the running time and quality of the solution finally obtained. If an algorithm takes more running time, then more partial solutions will be explored and better results will be achieved. Thus, we can consider an algorithm to be superior to another one if it is able to achieve a better result with less running time. All the experiments are conducted on a server with Quad-Core AMD Opteron(tm) Processor 8356,

128GB memory, running Centos 5.4. Since NP and BPG methods involve randomization, we repeat their experiments 10 times and retrieve the mean objective value as the result.

4.4.1 Performance of NP method

In the NP method, the random sampling probability p is important in the quality of the results. We first conduct a series of robustness analysis to explore the proper value of constant p. Since the best objective function value has higher probability to be selected, we only test the value from 0.7 to 1. We design the scenario with 2 QCs, 4 PMs, 3 YC Locations, and the number of jobs varied from 50 to 100. All the experiments are finished within 10 seconds.

Table 4.1 shows the numerical results of the NP method using different values of the sampling parameter p. We can observe that the NP method can generate better results with higher sampling probability p. A more visual comparison is shown in Figure 4.5, from which we can see the influence of different p values on the objective. It is obvious in most cases that when p is close to 1, the NP results dominate those with smaller p values in all scenarios, which is consistent with Shi (2000)'s findings. In their paper, the performance increases as p grows. The best result occurs when p is set to 0.99 (p=1 is not tested here).

Table 4.1 NP Results with Different parameter p values for 50-100 Jobs (2 QCs, 4 PMs, 3 YC Locations)

p\Jobs	50	60	70	80	90	100
0.7	134.494	171.749	195.697	232.471	262.368	292.559
0.72	136.85	166.639	195.968	222.589	258.487	288.401
0.74	137.011	165.235	190.606	225.769	259.726	288.626
0.76	137.537	167.275	195.147	222.031	258.681	280.561
0.78	131.505	168.542	189.956	219.664	255.181	272.219
0.8	131.536	160.129	187.273	222.094	255.965	279.671
0.82	128.769	160.078	185.379	214.462	245.964	273.635
0.84	128.623	157.954	189.551	214.083	255.85	277.779
0.86	125.017	157.417	177.656	214.186	243.176	265.375
0.88	122.769	155.483	184.212	203.974	242.256	262.671
0.9	123.309	147.47	173.401	198.186	244.885	257.191
0.92	117	147.625	169.993	202.056	235.5	260.45
0.94	118.968	144.026	163.521	195.677	227.636	250.379
0.96	113.768	138.58	162.152	188.857	219.511	241.103
0.98	109.252	130.52	151.971	176.808	208.109	226.88
1	107.305	126.264	145.505	171.689	193.838	211.514

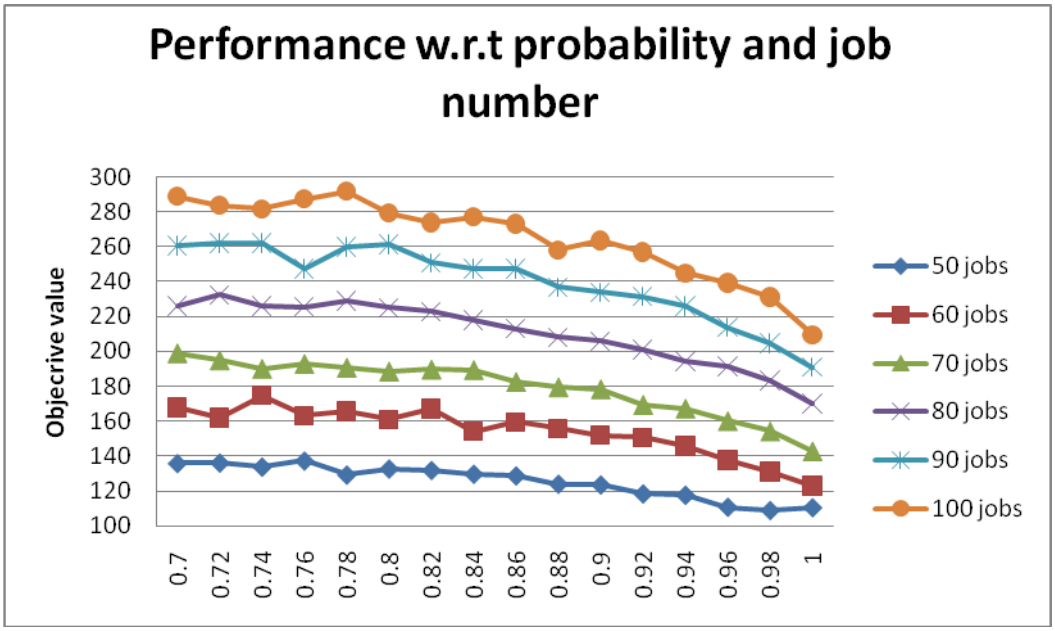

Figure 4.5 Parameter p's influence in NP with 50-100 Jobs (2 QCs, 4 PMs, 3 YC Locations)

Next we design a series of scenarios with 2 QCs, 3 YC Locations, and different PMs to test the trend of vehicle effect. We set the random sampling parameter p to be 0.98. The results of scenarios with different PM numbers and job numbers are plotted in Figure 4.6. We can observe that the larger the PM number is, the better the result is. As the number of vehicle increases from 4 to 8, the objective values decrease in all scenarios accordingly. However swe also can perceive that when the PM number increases above 5, the trend of objective value becomes quite stable.

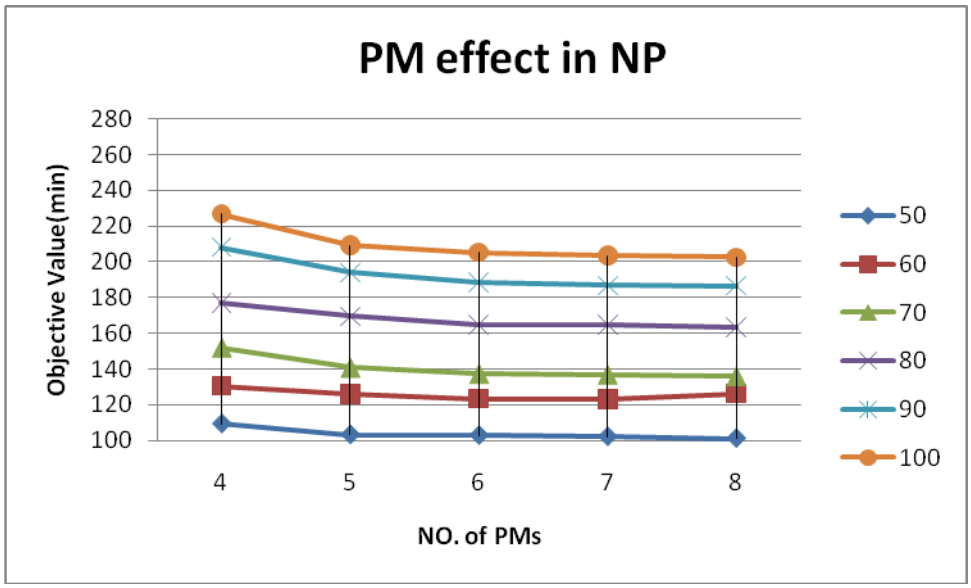

Figure 4.6 PM effect in NP with 50-100 Jobs (2 QCs, 4 PMs, 3 YC Locations with sampling probability p=98%)

4.4.2 BSG method results

We test the effect of buffer size in the BSG method. For a given scenario, we changed the buffer size from 1 to 100. A typical trend of this change is shown in Figure 4.7. We can observe that when buffer size is small, the increase of buffer size takes effect and the

objective value decreases dramatically and quickly reaches a local optimum. Later, the increase of buffer size does not help much. The reason is that buffer size grows linearly while search space grows exponentially and results in a large number of redundant partial solutions to be captured in the buffer. Hence, it is unwise to blindly increase the buffer size since larger buffer size takes longer computation time. Inspired by this fact, we focus on a buffer size of 40-60 to improve the solution in BSG. Another interesting phenomenon in this graph is that there is an embossment in the curve (when buffer size is 7) in Figure 4.7. In other words, larger buffer size cannot guarantee better solution. It can be explained that when buffer size is larger, much more search regions are generated in each iteration. A better solution may be abandoned when buffer size is large due to the masses of competitors, while this solution is preserved when buffer size is small.

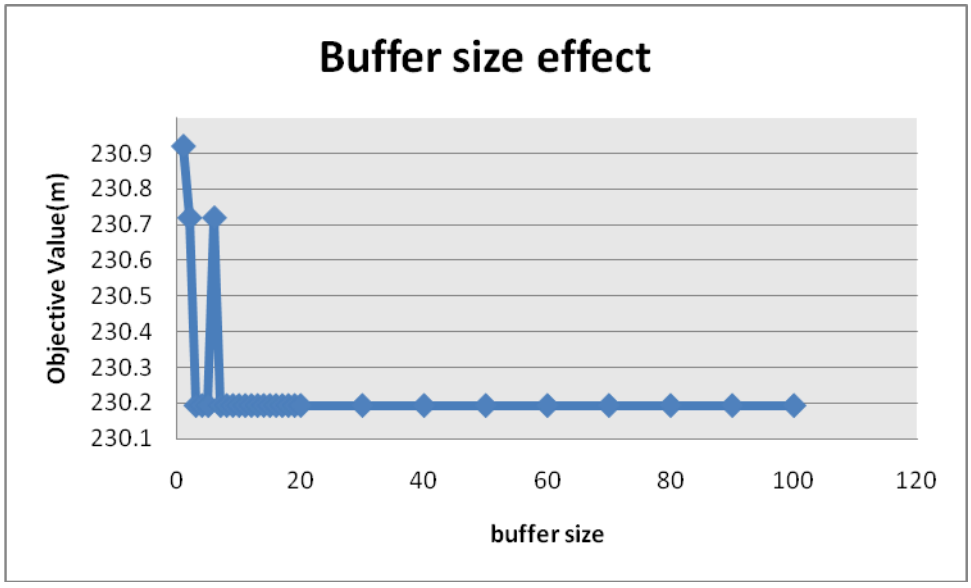

Figure 4.7 Robustness of buffer size (100 Jobs, 2 QCs, 3 YCs, 8 PMs)

4.4.3 Comparison between NP, BSG and BPG

In this set of experiments, we focus on buffer size 50 to assess the performance of the BPG method. We conduct the same experiment designs of 2 QCs, 3 YC locations with the number of jobs varying from 50 to 100, and the number of PMs varying from 4 to 8 to compare with the NP method and the BSG method. We compare with the best solutions of the NP method which set random sampling probability to 0.98. The comparison experiments' results are shown in Table 4.2.

Table 4.2 Comparison experiments results

	50	**60**	**70**	**80**	**90**	**100**
4 Trucks						
BPG	**106.267**	**119.883**	**139.94**	**166.269**	**186.062**	**203.618**
NP	107.305	126.264	145.505	171.689	193.838	211.514
BSG	110.004	130.214	151.516	179.262	201.157	217.028
5 Trucks						
BPG	102.947	**119.049**	135.142	**161.001**	**182.969**	**199.085**
NP	**102.091**	123.115	**133.328**	165.63	185.348	203.073
BSG	109.35	127.35	135.556	174.675	196.17	214.027
6 Trucks						
BPG	102.909	**119.073**	133.948	163.303	**178.397**	202.696
NP	**102.055**	120.488	**131.835**	**162.373**	178.398	**201.065**
BSG	109.35	127.35	135.107	170.686	194.208	218.647
7 Trucks						
BPG	102.744	**119.731**	135.004	**157.529**	181.767	**200.834**
NP	**102.082**	120.488	**133.821**	158.055	**178.398**	201.065
BSG	109.35	127.35	135.107	170.086	194.208	218.647
8 Trucks						
BPG	102.762	**119.092**	135.604	**157.21**	180.87	**200.845**
NP	**102.082**	120.488	**133.821**	158.055	**178.398**	201.065
BSG	109.35	127.35	135.107	170.086	194.208	218.647

Table 4.2 shows more directly that in most cases our method BPG gets best performance. Besides, we also observe that BPG gets close values to NP in the remaining cases. Note

that the NP results here are the best it can obtain with the perfect random sampling probability 0.98, and it would be much worse when this parameter value changes.

In meta heuristics, sophisticated methods improve the solutions while increasing the computation time as the emphasis is on performing a deep exploration of the most promising regions of the solution space. On the other hand, some simple methods can find a good but not best solution in a very fast speed such as our BSG method here. To capture this tradeoff between gain and cost, we propose a new assessment mechanism which takes the computation time factors into consideration. As shown in Figure 4.8 and Figure 4.9, the horizontal axis stands for computation time, and the vertical axis stands for solution quality which is the ratio of "Optimal" solution to experiment results. The pseudo "Optimal" can be any real number near but less than all the experiment results, which is used as a normalization factor. We plot two graphs to illustrate this quality vs. CPU time comparison. An algorithm is considered superior than another one if it is able to achieve a better result with less running time. In other words, it should be plotted at the left-top corner of the figure.

As shown in Figure 4.8 and Figure 4.9, the square shape represents the solution obtained from NP method, the diamond shape represents the solution of BSG, and the triangle shape represents that of BPG method. We plot every dot for each method by using different parameters (sampling probability in NP, and Buffer size in BSG & BPG) which result in different computation times. For these two graphs, we can observe that NP method relies on high sampling probability parameter value and runs longest time to find a solution; while the BSG method can generate a good solution in very limited time. In the

view of getting a better solution in the limited time, our BPG method performs best over the other two (BPG is always in the top left corner), especially in the large case when job numbers and resource numbers increase.

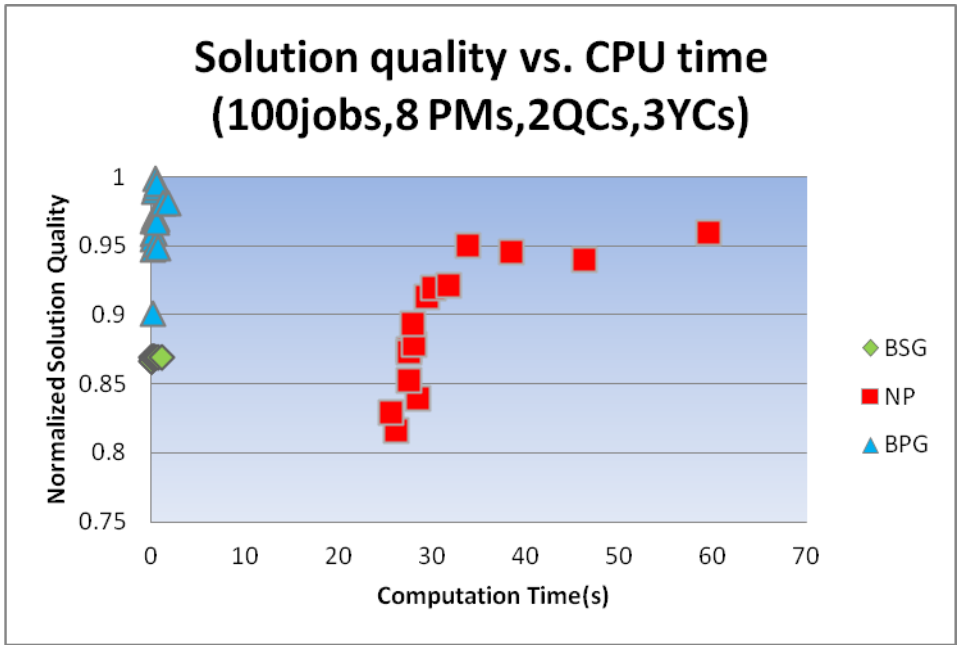

Figure 4.8 Best Case Comparison results (100 Jobs, 8 PMs, 2 QCs, 3 YCs)

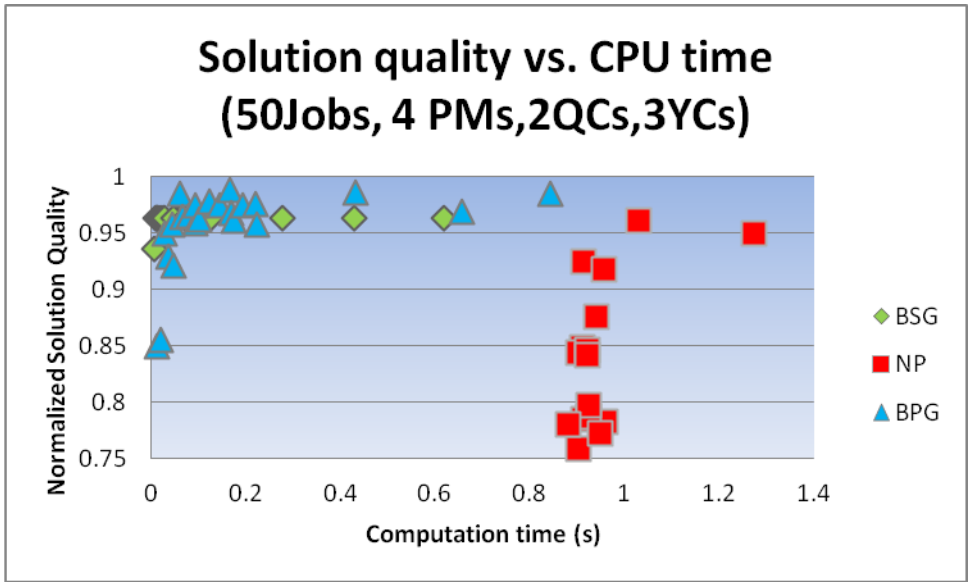

Figure 4.9 Worst Case Comparison results (50 Jobs, 4 PMs, 2 QCs, 3 YCs)

4.5 Summary

In this chapter, we discuss on how to assign the container jobs to PMs as well as assignment of discharging jobs to potential yard locations. Three methods are proposed to tackle this integrated problem and they are NP, BSG and BPG methods. We have shown that the NP method can solve this problem efficiently by using proper parameters; moreover BSG can generate a feasible solution very quickly which can used in real time dispatching, while the improved BPG method dominates the other two in getting a good quality solution in a limited time. The contributions to this problem are:

1. Three strategies are developed to improve the terminal productivity by integrating dispatching and location problem. Unlike other methods, these strategies can explore deeply in the promising region.
2. An efficient BPG approach is proposed to solve this integrated problem. Unlike other two phase cyclic methods, BPG builds solutions gradually and adapts the history information in each stage.
3. A novel assessment scheme is proposed which evaluates the performance in a different angle, taking CPU time and solution quality into consideration.

5 The Integrated Simulation Platform for Real Time Dispatching

In real time dispatching, port operators usually use simple rules as they need to make decisions in a short period of time. Some researchers develop more complicated rules by capturing more information. However, in practice it is not easy to know the effectiveness of different rules under different scenarios, since the system can be highly dynamic and stochastic. This chapter aims to present a simulation framework which is able to evaluate the performance of different dispatching rules. This framework consists of two main modules: dispatching module and simulation module. One of the most significant contributions of this framework is that the two modules can communicate with each other effectively regardless of using simple rules or complex heuristics. The dispatching module can be any external sophisticated heuristic model or simple rule; the simulation module here is developed based on a platform named MicroPort. In this chapter, we conduct two series of experiments to evaluate different real time dispatching methods. One of them is to analyze the performance of simple rules and some look ahead strategies; the other is to evaluate the performance of a dispatching heuristic based on the solving of the complicated optimization models.

5.1 Introduction

For real time dispatching, we need to make decisions based on the most updated information, and this usually cannot be considered directly during the planning phase. In

practice, port planners usually use simple greedy rules to do real time dispatching. On the other hand, some researchers propose more complicated rules which can capture more information based on the look ahead planning. In order to evaluate the effectiveness of these rules, we need to use simulation. However, it might not be easy for us to code the rules especially those complicated rules into current commercial simulation software because these complicated rules might involve solving an optimization model.

In this chapter, we introduce a simulation platform named Microport for evaluating the real time dispatching rules for a container terminal. From the perspective of software, the Microport uses separated layers to provide many useful functions which allow flexible communication between different modules; from the perspective of simulation, this software can simulate interactions among all equipment in a container terminal by the proposed multi-agent modeling method. The Microport combined with our dispatching model can provide an efficient simulation platform for evaluating real time dispatching rules. This platform has two advantages. Firstly, it enables dialogue between simulation models and dispatching models, in which we can try different heuristics which range from simple heuristics to complicated heuristics and evaluate them easily; Secondly, it is possible to generate different terminal layouts to test different algorithms in different scales by just easily changing the terminal layout parameters.

5.2 Real time Simulation Platform

The essence of real-time lies in the fact that decisions are made as and when they are required. For the real time dispatching problem, researchers have developed various rules

ranging from simple greedy heuristics to complicated mathematical models. However, it is quite challenging to evaluate the rules, especially the complex algorithms via current commercial simulation software due to the inefficient communication between different modules and software. This study is motivated by an actual port's real time dispatching problem, and our aim is to facilitate the simulation process and help evaluate different rules for real time dispatching. In this chapter, we present a real time simulation platform and discuss its special features.

5.2.1 General Framework

The general framework of the platform can be illustrated in Figure 5.1. The framework comprises of two main modules: a dispatching module and a simulation module. The dispatching module can be an optimization model which is able to determine the optimal assignment for yard cranes and vehicles given the working list of quay cranes. The simulation module is developed to address two main issues in this dispatching problem. Firstly, the updated states generated by the simulation module will be passed to the dispatching module to determine the real time dispatching decisions; secondly, it is flexible to test and evaluate different dispatching models in different layouts and different scenarios. This platform can implement simple rules as what other simulation software can do. Moreover, the most important feature of this platform is that we can easily implement and evaluate different complicated heuristics algorithms using this platform. These two modules work together. Once the dispatching model gives a solution (in our problem, it is the job sequence on the vehicle) based on the given information, the simulation module simulates the first job in the solution; after that, the dispatching model

will revise the solution based on the new realized information. These procedures repeat until all the jobs are finished or until the simulation termination conditions are met.

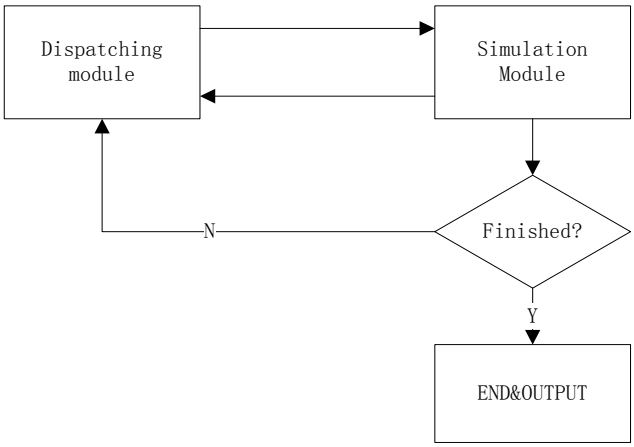

Figure 5.1 The flowchart of the platform

5.2.2 Simulation Platform Features

We use multi-agent systems to model the container terminal operations in which each equipment and decision making is represented by a specific agent. The proposed multi-agent system has a main thread and many agent coroutines. The main thread can switch the system between serial mode and concurrent mode. In the serial mode, the system halts all agents and executes a sequence of procedures for central controlling. These procedures make overall decisions based on current system status. When the system goes into concurrent mode, distributed agent coroutines will be activated and they interact with each other in the context of overall decisions made in previous serial mode. For instance, there are dozens of yard trucks traveling in the yard and interacting with other equipment. The job sequence in each yard truck is scheduled by the yard truck scheduler in the control center of the terminal. To model this subsystem in MicroPort, every yard truck can be represented as an agent coroutine and the yard truck scheduler can be treated as a central

controlling procedure. The yard truck scheduler will be regularly triggered or triggered by events to pause all of the agent coroutines to make plans for yard trucks. Then the simulation will be resumed and all of the agent coroutines are activated simultaneously to continue the yard truck behaviors.

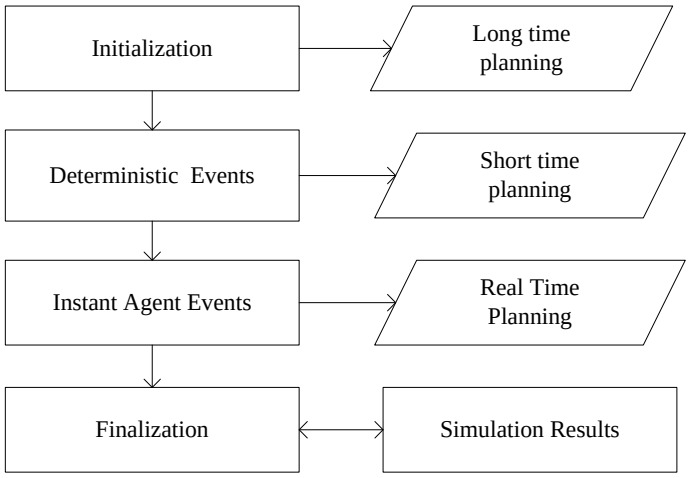

Figure 5.2 Simulation and Terminal Decision Planning

The special software provides a flexible simulation modeling environment. In MicroPort, equipment are modeled as agents and interactions between them are modeled as concurrent agents' events. Decision processes are modeled as serial procedures shown in Figure 5.2. The simulation starts at repeatedly vessel arriving. This event will trigger a series of decision processes. These processes make all of the long-term, short-term and real-time decisions followed by the terminal equipment. Terminal equipment use the schedules set by decisions as prior rules and interact with each other to deliver containers. They will trigger back to decision processes when they reach certain states. The long time planning for our problem refers to berth allocation, quay crane assignment; quay crane scheduling; the short time and real time dispatching here consist of yard crane dispatch

and yard truck dispatch. These planning algorithms are stored externally and loaded dynamically during runtime. APIs provided in the extension layer facilitate loading activities.

1) Communication Capability

When it is triggered from other decision making procedures, the short-term deterministic yard truck dispatch module will be executed. Because of the ever changing environment of the yard, this module only considers limited future tasks in the quay crane task lists and is always incorporating the short-term deterministic yard crane dispatch procedure to make more global optimized decisions. When the yard truck dispatch procedure is triggered from waiting or moving yard trucks, the real-time decisions making unit should respond immediately. These real-time decisions are based on short-term planning results and can have many detailed instructions including destination and routing information. This procedure is not only suited for yard trucks but also can be adapted for other traveler equipment in the yard. Different user specified algorithms can be plugged in without considering the synchronization.

2) Collision Avoidance

Because quay cranes and yard cranes are designated to a single lane along their moving path, they cannot perform cross gantry travel. Hence the minimum spacing between quay cranes and yard cranes is mainly controlled by the decision processes. The minimum spacing for quay cranes and yard cranes can be set before simulation. In simulation when two quay cranes or yard cranes go too close, the corresponding decision process will be triggered to rearrange the two equipment schedules and current moving destinations. The

interference between two yard trucks can cause collisions. In simulation the autonomous agents are responsible for avoiding collisions. If yard trucks are controlled by computer like AGVs in some terminals, the interference should also be carefully modeled in the yard truck dispatch procedure.

3) Deadlock Avoidance

In software layers, MicroPort has already eliminated deadlocks among agents by using collaborative coroutines. But in logic aspect, deadlocks can also happen when several equipment are in a waiting cycle. Quay cranes' schedules are almost fixed once they have been decided. Thus it becomes more important to control yard trucks and yard cranes in order to avoid deadlocks. In real-time decision making units, MicroPort only considers those surely appearing equipment, which ensures that the waiting equipment is not waiting for other equipment that potentially never shows up.

5.3 Dispatching Module

In this section, we focus on the real time and short time planning which consists of yard crane and yard truck dispatching. We introduce two series of dispatching rules: simple rules and complex heuristic models. The former series is embodied directly in MicroPort and the latter one is individual optimization models coded in C++ and calling MicroPort to simulate the real situations.

5.3.1 Simple Rules

These are generally basic rules used for the immediate dispatching of a job to a single-load truck based on current local information such as current vehicle position, whether the resources are busy or idle at the current time, etc. In these basic strategies, simple decisions are made when events occur based on current available information. In this section, we analyze two categories of simple rules: one is typical greedy rules and the second one is simple predictive look ahead rules.

5.3.1.1 Greedy Rules

5.3.1.1.1 Task-Initiated Strategies

Task initiated strategies, also known as transportation-order-driven strategies (Günther et al. (2006)), refer to the assignment of jobs to vehicles that is task-driven. Here we introduce 2 task-initiated strategies: nearest vehicle (NV) and least utilized vehicle (LUV) strategies.

- Nearest idle vehicle (NV) - When a particular quay crane has successfully transferred a job to a vehicle, the next job will be assigned to the **nearest** idle vehicle at that point in time. However, as pointed out by Günther et al (2006), this rule may discriminate against vehicles that are far away from active quay cranes and result in the disproportionate use of vehicles. To counter this, we introduce the Least Utilized Vehicle (LUV) strategy.
- Least Utilized Vehicle (LUV) – The utilization of a vehicle is defined by the time in which the vehicle is in operation / the total port operation time period. In this case, the quay crane will assign a job to a vehicle that is the most under-utilized.

However, the downside is that a particular vehicle may be located at some distance away and will end up arriving late.

5.3.1.1.2 Vehicle-Initiated Strategies

Unlike task-initiated strategies, vehicle initiated rules are called when a vehicle ends its current job and looks for a job instead of going straight to idleness. In other words, the events which call for these decision rules are vehicle-driven. Generally, vehicle-initiated strategies keep the vehicles on the move instead of going straight to idleness. Here we introduce two vehicle-driven rules:

- Nearest Quay Crane – In this decision rule, the vehicle selects the next job on the quay crane that is closest to it. This is to minimize the travelling time to a quay crane so as to collect the respective job in time. However, this strategy may create a highly non-uniform vehicle-to-QC allocation and result in extremely high waiting time when all the vehicles select disproportionately to jobs of a particular QC while neglecting the more critical ones. This is a possible worst case scenario that can potentially increase the quay crane waiting time significantly and ultimately affect the vessel turnover rates. Hence, we propose a second strategy to handle this issue; the Most Critical Quay Crane Strategy (MC).

- Most Critical Quay Crane (MC) – Each quay crane has 2 lists comprising of assigned jobs and unassigned jobs. The criticality of a quay crane can be reflected by the number of assigned jobs, because it signifies the number of vehicles that are waiting or travelling to that quay crane. The low number of the assigned jobs indicates that the quay crane is likely to experience a no-show from any vehicles when the job is ready for collection.

5.3.1.1.3 Crane-Initiated Strategy

This strategy involves that of a crane-initiated one, which fixes vehicles to cranes. The nature of this strategy is First-Come-First-Serve (FCFS), where vehicles, upon ending their jobs, proceed back to their assigned crane to collect the next task. One inherent risk of fixing vehicles to a quay crane is the possible scenario that these vehicles may be assigned jobs that are far away, resulting in significant crane waiting time, while vehicles to a neighbor quay crane remain idle in their wait to for their respective task collections.

5.3.1.2 Predictive Rules

Real-time decisions should incorporate uncertainties which are tough to foresee at the start of the planning. Hence, we introduce the predictive look-ahead strategy that updates and overwrites previous vehicle assignments as new information is being incorporated. Look-ahead dispatching takes into account of limited information at a particular point in time to produce a temporary predictive schedule for vehicles. These can be seen as a series of static problems at several points in time. When an event occurs, new information will be incorporated into the previous static problem, forming an updated one which is to be solved again. These look ahead rules consist of initial solution and local search. The initial solution is a feasible starting point for the relevant heuristic. Local search involves consideration of the current solution's neighbors, which are solutions in close proximity of the current solution. In the next section, several heuristics will be employed to contrast and compare the different approaches towards forming the initial solution as well as conducting the local search. These include Hill Climber, Simulated Annealing and GRASP.

5.3.1.2.1 Hill Climber

Hill climber, as the name suggests, works with the principle that the climber takes the next step that progresses him higher up till he reaches the peak. This algorithm starts with a random initial solution and searches the neighborhood for the next better solution. However, the hill climber strictly selects only the first and better neighbor to progress onwards. The 'better' neighbor is defined by a close solution with respect to the current one that produces a better result, which in this case, refers to lower costs related to crane and vehicle waiting times. Simple as it is, the hill climber serves to provide a quick and simple solution. However, the problem is that randomizing the initial solution will create poor solutions because the solution span as mentioned earlier is so large. Furthermore, strictly selecting the first and better neighbor to progress on may subject the heuristic to the risk of getting stuck at a local optimum quickly.

In our problem, the first step here is to create an initial solution based on randomization, which is to assign vehicles to assignments in the Op_list where tasks are listed based on their due times. Given the instance of a look-ahead of 2 jobs scenario with all 10 quay cranes operating, the size of Op_list would be 20. The local search will comprise of 2 aspects: swapping of vehicle assignments in the list as well as consideration of vehicles which are not assigned to any tasks in the current solution.

5.3.1.2.2 Simulated Annealing

Another way of avoiding local optimal solutions is to adopt another heuristic by the name of simulated annealing. The essence of this heuristic lies in its local search which attempts to avoid getting stuck at local optimum early in the run by providing a certain acceptance

probability to a neighbor that performs poorly compared to the current solution. This probability is incorporated from the theory of thermodynamics: $p(\delta, ti) = \exp(-\delta / ti)$, where δ is the difference between the value of the lower neighbor and the current solution. ti refers to time. The probability typically decreases with respect to time and the size of the shortfall δ. Similar to the Hill Climber Algorithm, the initial solution will be based on a random assignment of vehicles to the tasks in the Operation_list (Op_list). However, the only difference here will be the inclusion of the acceptance probability should the heuristic encounter a neighbor which does not perform as well.

5.3.1.2.3 GRASP

The greedy randomized adaptive search procedure (also known as GRASP) is first introduced in Feo and Resende (1989). GRASP is a more advanced heuristic to solve combinatorial optimization problems. ***Greedy*** Algorithm is used here to create an initial solution which will minimize the waiting time of quay cranes. ***Randomization*** here introduces the idea of a ***Restricted Candidate List (RCL)*** whereby each member of the RCL is within a stipulated allowance β%, where $0 < \beta < 100$, in terms of performance. In the case of a strict greedy rule, $\beta = 0$. Within this RCL, the selected candidate will be randomized with the probability of 1/s, where s is the size of the RCL and is dynamic. Under the ***adaptive*** condition, the selection criterion for the greedy solution 'adapts' to the respective scenario and is therefore dynamic. Finally, the ***search*** element involves local search within the neighborhood to find a better solution. ***Procedures*** incorporate the sequences that involve the creation of the initial greedy solution right to the local search, till a good solution has been found.

The Greedy, Random and Adaptive functions work together to construct an initial solution for the assignment problem. In forming the initial solution, the key step lies in the setup of the Adaptive function for the selection of the candidates from the RCL. The RCL candidates will comprise of certain selected vehicles which result in one of the lowest QC waiting time for a particular task in the Operation_list where tasks are listed based on their due times. Then the list of vehicles is sorted according to the lowest crane waiting time.

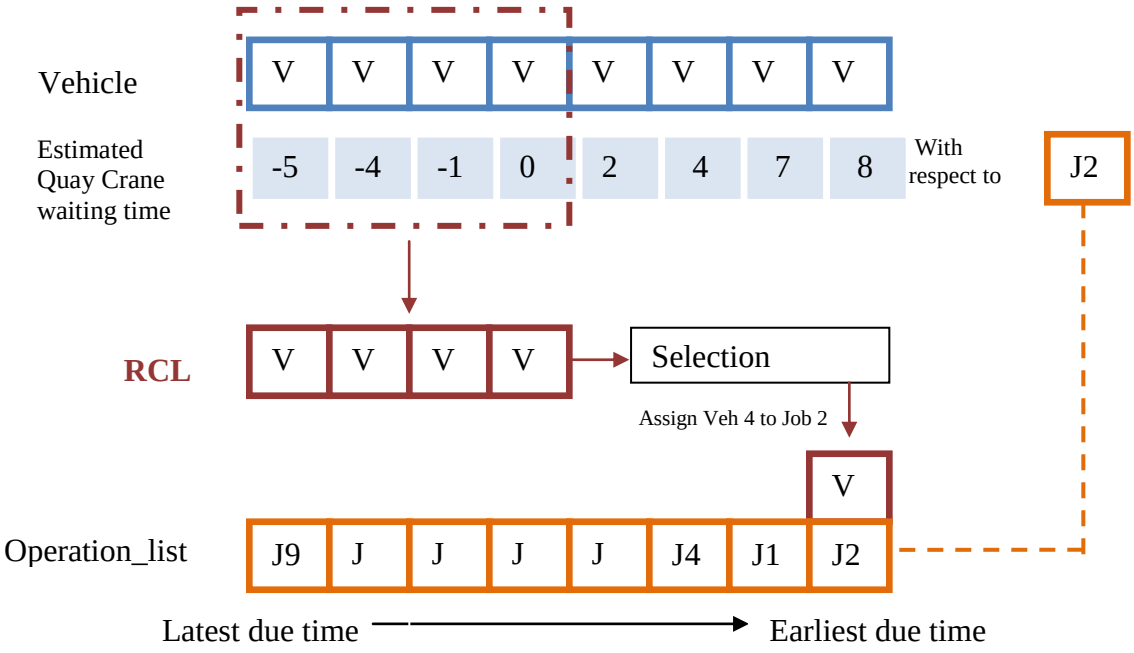

Figure 5.3 Construction of an initial solution using GRASP

In this scenario (see Figure 5.3), we firstly estimate the resulting crane waiting time should the respective vehicle be assigned to Job 2 for all the vehicles in the vehicle list. A negative crane waiting time denotes the vehicle arriving before the job is due, which means positive vehicle waiting time. Depending on the specified parameter for the selection of the RCL candidates, if in this case the allowance from the best performing candidate is 5, then vehicles within the range of [-5,0] in terms of crane waiting time will

be entered into the RCL. In this case, this will include V2, V4, V1 and V3. As for the final selection to the assignment of Job 2, since we are indifferent towards negative values, we can just safely select randomly from any of the four vehicles and assign to J2. Next, we proceed on to Job 1 which is the next earliest job due and the process is once again repeated.

However, the selection criterion does not remain the same for all scenarios. Hence, it is essential to identify the various scenarios in which the selection criterion will change.

1) Scenario 1: As presented earlier, if the entire RCL consists of vehicles that result in zero crane waiting time, randomly select any of them.
2) Scenario 2: If the RCL consists of some vehicles (more than 1) that result in some degree of waiting time as well as vehicles with no waiting time, select randomly from the vehicles that only result in no crane waiting time.
3) Scenario 3: If only 1 candidate results in no crane waiting time, then strictly select it.
4) Scenario 4: If the RCL comprises of vehicles that result in some degree of crane waiting time, select randomly.

The idea here is clear and consistent in all 4 scenarios. The priority is given towards the assignment of vehicles with the ultimate primary aim of minimizing crane waiting time. The next step is to conduct a local search. There are 2 aims of the local search. The first is again to check if the overall QC waiting time could be further reduced. The second aim is to reduce the vehicle waiting time without increasing the QC waiting time. Hence, neighbor acceptance is only allowed under two conditions; when both crane and vehicle waiting times are reduced, or when vehicle waiting time is improved without affecting that

of the quay cranes. There are basically 2 types of searches; the first being similar to that of the Hill Climber while the other being similar to that of Simulated Annealing.

5.3.2 Complicated Models(GA-MCF Model)

Other than the above simple dispatching rules, we also provide more complex optimization heuristics with individual packages. This complicated optimization model can have a dialogue with the simulation platform. In other words, this optimization model will try to solve the integrated dispatching problem and the simulation platform provides the real dynamic environment and evaluates the optimization heuristic. In the software aspect, they are two packaging systems which communicated with each other dynamically. In this thesis, we will choose one novel and sophisticated optimization model in Lee.et.al (2010). The method is based on genetic algorithm (GA) and minimum cost flow (MCF) network model (see Chapter 3.2.3). However, we made some changes to the GA-MCF model here to facilitate communication between the optimization model and the simulation module by considering the beginning yard truck location.

The goal of this minimum cost flow (MCF) model is to find a schedule that will minimize the impact of delays and maximize the utilization of the vehicles (Cheng 2005). However in our work, we intend to use MCF as a means to generate good PM (truck) sequence. Our model can also be viewed as a directed graph G (*V, A*) where *V* denotes the set of nodes and *A* denotes the set of arcs. All container jobs in set H(contains both discharging and loading jobs) and the dummy ending job E, as well as each PM truck are represented as nodes in *G*. If two jobs are served by the same vehicle, there is a directed arc connecting

them. We need to determine *m* routes from PM nodes to node *E* when there are *m* vehicles deployed to serve jobs. The cost of the arc is represented by the deviation of the ready times and our objective is to minimize the overall network cost.

Let X_{ij} represent the flow on arc (i, j) and C_{ij} be the cost parameter, the can be formulated as follows.

The model can be formulated as follows.

MCF Model

Minimize $$\sum_{i\in J}\sum_{j\in J} C_{ij}X_{ij} \quad (5.1)$$

Subject to:

$$\sum_{i\in V:(m,i)\in A} X_{mi} = 1, \text{ for } m \in M \quad (5.2)$$

$$\sum_{i\in V:(i,j)\in A} X_{ij} = 1, \text{ for } j \in H \quad (5.3)$$

$$\sum_{i\in V:(j,i)\in A} X_{ji} = 1, \text{ for } j \in H \quad (5.4)$$

$$\sum_{i\in V:(i,E)\in A} X_{iE} = m \quad (5.5)$$

$$X_{ij} \in [0,1], \text{ for } (i,j) \in A \quad (5.6)$$

Equation (5.1) states the objective which minimizes the total cost of the flow. Constraints (5.2), (5.3), (5.4) and (5.5) are the flow conservation equations for the *m* vehicles. Constraints (5.6) limit the flow to not more than 1.

Let t_i be the ready time for the QC to pick up (for discharging) or drop off (for loading) containers for job *i*. Let t_{ij} denotes the time interval between the time when the PM starts to do the job at QC for job *i* and the time that it is ready to perform job *j* at the QC location for job *j*.

Hence, $t_i + t_{ij}$ is the time that the PM arrives at the QC which is assigned to process job *j*, and C_{ij} which measures the deviation of the ready time for job *j* after serving job *i* is given as follows.

$$C_{ij} = \begin{cases} t_i + t_{ij} - t_j & \text{if } t_i + t_{ij} - t_j \geq 0 \\ \gamma(t_j - t_i - t_{ij}) & \text{otherwise} \end{cases}$$

where $\gamma > 0$ and it is a constant.

Note that γ is a parameter that gives the relative weight between being early and being late. Being late would cause the QC to wait while being early, will not only cause the PM to wait, but also may result in infeasibility due to the fact that the QC sequence is violated. We have tuned the parameter γ during the numerical runs.

Similarly, let t_m be the ready time for PM m. Let t_{mi} denotes the time interval between the time for PM m being ready to start and the time that it is ready to perform job *i* at the QC location. We have the cost between PM node m and job node i, which is to capture the state of the PM at the time of deployment.

$$C_{mi} = \begin{cases} t_m + t_{mi} - t_i & \text{if } t_m + t_{mi} - t_i \geq 0 \\ \gamma(t_i - t_m - t_{mi}) & \text{otherwise} \end{cases}$$

The arcs from all container job nodes to the dummy ending node *E* are assigned with zero cost, i.e., C_{iE}=0 for all *i*.

Choosing the "right" ready times is important because some ready times may give infeasible PM sequences while some ready times may give good or even optimal PM sequences. We prove that there exists a set of ready times which will give the optimal PM sequence. Hence the MCF model is used in two different ways in our proposed heuristic. Firstly, it is used to generate the initial PM sequence given initial ready times for all the jobs. Secondly, it is used as a decoder for the GA approach when the chromosome is represented by ready times. The procedure of the proposed GA is shown in Figure 3.9. Compared with the model in chapter 3, we add initial vehicle time to capture the real vehicle location information.

To implement this complex heuristic algorithm, we build a connection between this optimization model and the simulation module. It begins from the optimization model to generate a solution, and then the simulation starts to simulate the first job. After that the optimization model is used to generate a new solution for the remaining jobs based on current information and then the simulation starts again. These procedures repeat until all the jobs have been simulated. This simulation platform enables to evaluate complex algorithms to be evaluated, which is the most significant difference from other simulation software. Through this platform, the optimization and simulation can communicate and share current information to help real time dispatching.

5.4 Case Study for Simple Rules

Simulation will be adopted to compare the performances of the various dispatching strategies. This simulation study will be conducted based on a base model ran by our simulation software MicroPort. The nature of this model is that of a discrete-event driven one, whereby the state of the system changes when an event occurs at an instant in time.

Below are some of the model assumptions:

1) The layout of the terminal comprises of 45 storage blocks, with 10 QCs and 45 YCs, 1 YC for each block (shown in Figure 5.4).
2) The total number of vehicles in the base case is 30 and it will be varied between 30, 35, 40 and 45 to see the impact on the port performance. Each vehicle will only be able to carry a single load.
3) Berths are randomly assigned to incoming vessels with an inter-arrival time uniformly distributed at 25 hours.
4) Terminal operations only focus on discharging operations; containers are offloaded from vessels and stored at the yards, no containers are loaded on the vessel.
5) Quay Crane and Yard Crane processing times are deterministic at 1/30 hours, while vehicle speed is deterministic at 12 km / hour.
6) Vehicle routing will be taken care of by the simulation software which includes traffic conditions. This means that the vehicle path to a particular destination will be the shortest (optimal) one.
7) All Equipment are assumed to have infinite life-span with no disruptions to the operations with regards to accidents or breakdowns.

For a given set of parameters, each simulation run was conducted for approximately 700 hours, based on considerations for the warm-up periods as well as the general stability of the simulation model.

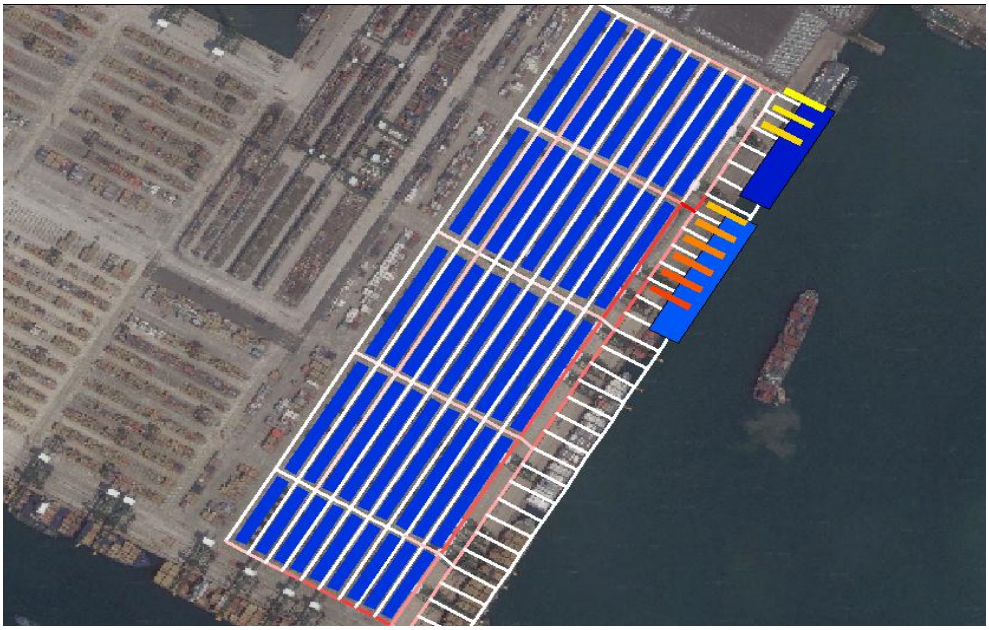

Figure 5.4 Screenshot of the case study simulation

5.4.1 Performances of greedy dispatching strategies

We compare and analyze the performance measures for each of the dispatching strategies. Four performance measures will be used here: **QC lifts per hour**, **QC efficiency** (total non-waiting operation time for QC / total operation time), **average vehicle waiting proportion** (time spent waiting at QC / total simulation time) and **average vehicle utilization** (time period in which vehicle is in operation (including waiting time) / total simulation time). As mentioned, QC lifts per hour is the primary gauge for the vessel turnover time. However, a more accurate measure would be the QC efficiency because it accurately measures the amount of time the QC is effectively in operation.

Table 5.1 Results of greedy dispatching strategies

	Algorithm	**Average Quay Crane Lifts per hour**	**Average Quay Crane efficiency (non-waiting proportion)**	**Vehicle waiting proportion (at QC side only)**	**Average Vehicle Utilization**
Task Initiated	Base Case random	19.91	0.75	0.0545	0.519
	Nearest Vehicle (NV)	20.20	0.78	0.0522	0.624
	Least Utilized Vehicle (LUV)	19.00	0.73	0.029	0.592
Vehicle Initiated	Base Case Random	20.84	0.76	0.284	0.997
	Most Critical QC	25.30	0.86	0.169	0.997
	Nearest QC	17.32	0.69	0.787	0.997
Crane Initiated (FCFS)	Crane Initiated (FCFS)	22.02	0.77	0.322	0.888

We measure the individual performance of each of the dispatching strategies under task-initiated, vehicle-initiated and crane-initiated ones. For a given set of parameters mentioned above, each simulation run was conducted for approximately 700 hours. From Table 5.1, we can see that the best performing strategy is the Most Critical QC strategy, relative to the base case. It dominates because of its ability to identify which quay crane is in need of vehicle allocation. The Nearest QC algorithm on the other hand, performed relatively poorly. The crane-initiated algorithm, where vehicles are assigned to a quay crane on a first come first served (FCFS) basis, in general performed much better than the majority in the terms of quay crane productivity.

For the vehicle–initiated strategies, the large deviation from the base case in quay crane efficiency results clearly reflects their huge impact on the performance of the port. Intuitively, it can be expected because the vehicle-initiated strategies are likely to be exposed to more detrimental scenarios than the task-initiated algorithms. For instance, in adopting the nearest QC strategy, there is a likelihood that vehicles ignore cranes that are in urgent need of vehicles, due to their greedy nature for proximity to the cranes. In other words, the vehicle may end up selecting quay cranes based on proximity of distance, resulting in a negative impact on the port.

Task-initiated algorithms logically results in considerably lower vehicle waiting proportion as compared to vehicle-initiated ones because the task that is assigned to the next vehicle is just after the one that is currently being processed. The nearest QC algorithm is of concern in this case because it has a predominantly-high vehicle waiting proportion time of nearly 0.8. One explanation for this is that vehicles might have selected jobs from a quay crane which was already experiencing a long queue of vehicles. However, despite the significant waiting times, vehicle utilization is maximized under vehicle-initiated algorithms; which is logical due to the fact that vehicles are the ones that drive the events that call for the dispatching algorithms.

5.4.2 Performances of look-ahead heuristics

We design this group of experiments to evaluate the performance of different look-ahead dispatching strategies, such as GRASP, hill climber, highest hill climber and simulated annealing. The experimental environment is the same as before. The simulation results are compared with the base random case as shown in Table 5.2.

Table 5.2 Results of look-ahead dispatching strategies

	Average Quay Crane Lifts per hour	Average Quay Crane efficiency (non-waiting proportion)	Vehicle waiting proportion (at QC side only)	Average Vehicle Utilization
Base Case	22.26	0.77	0.324	0.946
GRASP	26.26	0.87	0.195	0.995
Hill Climber	15.48	0.53	0.344	0.966
Highest Hill Climber	19.93	0.71	0.291	0.995
Simulated Annealing	20.98	0.75	0.332	0.992

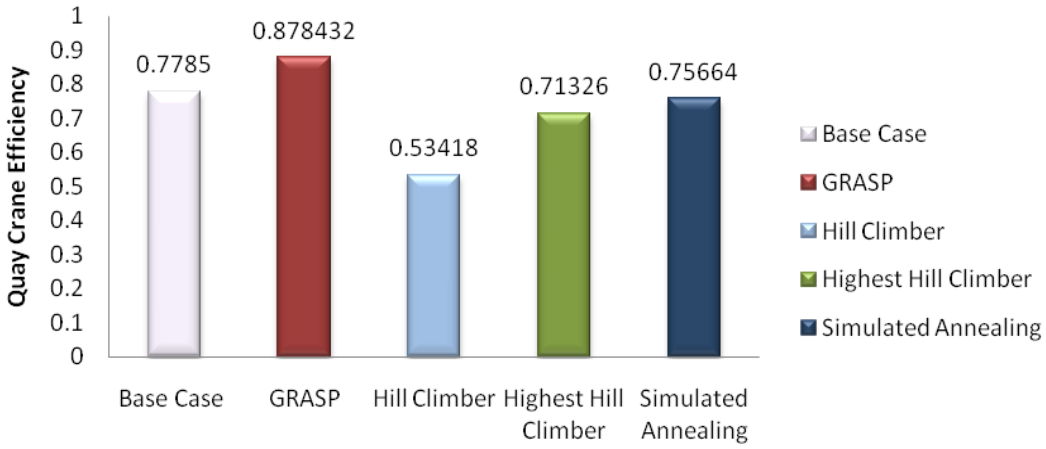

Figure 5.5 Comparison of algorithms for look-ahead strategy

It is shown in Figure 5.5 that GRASP has outperformed all other algorithms in quay crane efficiency. In general, the random-based algorithms such as hill climber take approximately 2 to 3 times longer to run as compared to GRASP. With just a simple implementation, GRASP has demonstrated itself to be an effective algorithm in maximizing quay crane efficiency. Both the hill climber and Simulated Annealing algorithms performed worse than the base case, which basically assigns the first job to the

first vehicle in a First-in-First-Out (FIFO) fashion. This clearly discourages randomization in an extremely large solution set.

5.4.3 Comparison with greedy and look ahead strategies

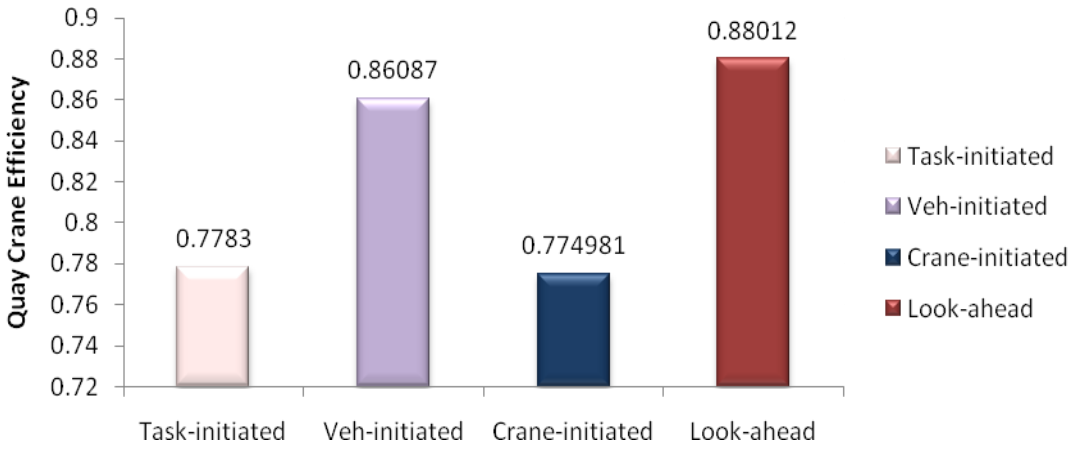

Figure 5.6 Comparison of the best greedy versus look-ahead strategies GRASP (QC Efficiency)

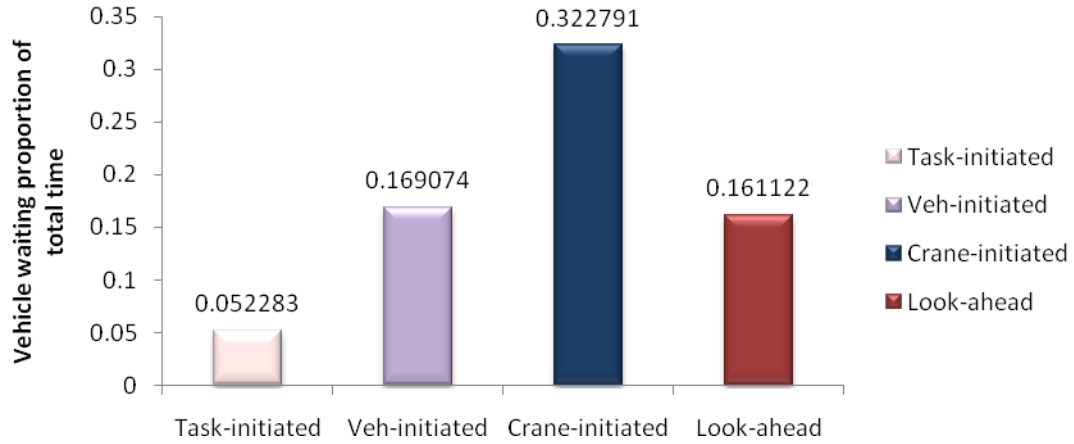

Figure 5.7 Comparison of the best greedy versus look-ahead strategies GRASP (Vehicle waiting proportion)

We compare the best strategy of each category in terms of Quay Crane Efficiency Performance (Figure 5.6) and Vehicle waiting Proportion (Figure 5.7) to see how they

perform among each other. Overall, GRASP performs the best in terms of maximizing quay crane efficiency. At the same time, it maintains a reasonably low proportion of vehicle waiting time. Another strategy which has performed reasonably well is the Most Critical QC strategy, which is the best performing vehicle-initiated strategy. What both strategies have in common is that they both address the underlying problem directly. The most Critical QC strategy checks for the QC which is in need of vehicles and assigns vehicles to them. As for GRASP, intuitively this has a lot to do with the structure and nature of the algorithm as well as the assignment problem itself. As the results have shown, greedy algorithms are not sufficient enough in producing a good solution.

5.5 Numerical Experiments for Complicated Models

One of the most advantages for our simulation platform is that it enables dialogue with external optimization models. It can communicate flexibly with complicated optimization dispatching models, which is significantly different from other simulation software. In this section, we conduct experiments of complicated GA-MCF heuristics in our simulation platform to test the real time effectiveness.

In our MCF model, we have PM nodes in the graph to capture the state of the PM at the time of deployment. Thus we can conduct 1 job look ahead simulation to capture the real time information. Once the optimization model gets the solution, the simulation just runs the first job, then we re-run the optimization model to get the next assignment and let the simulation run this solution based on the real situations. It is an iterative communication

between the simulation module and the dispatching module, which tries to mimic the real environment.

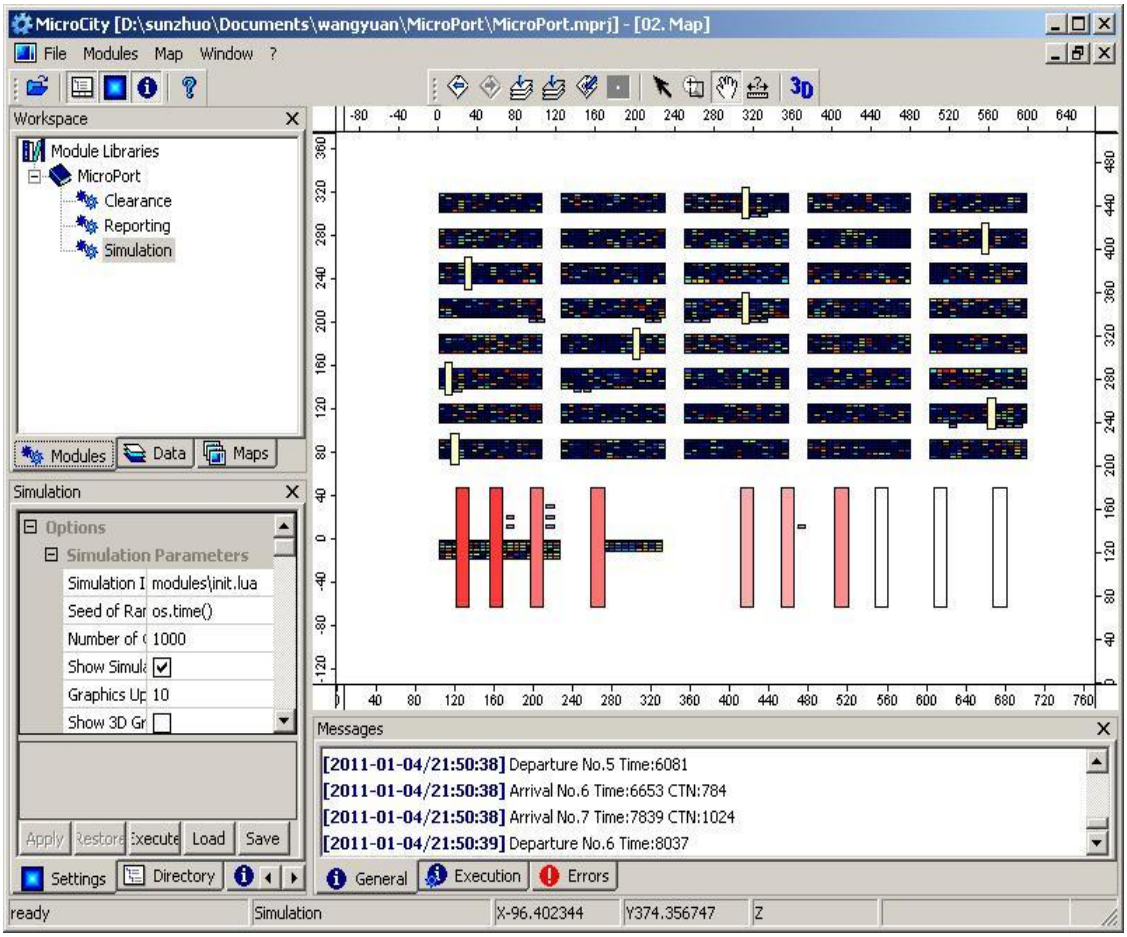

Figure 5.8 A screenshot of an application of Microport

To test the practicability of our simulation platform, a transshipment container terminal has been modeled. The geographical layout of this terminal is shown in Figure 5.8. We test 10 QCs, 8 YCs situations in order to compare results with different simulation methods. Other parameters are chosen based on major settings of a transshipment container terminal in Singapore. Resource statuses can be immediately presented during the simulation execution (Figure 5.8). This information is used to trace the activities of

quay cranes, yard cranes and yard trucks, and calibrate the simulation model. Table 5.3 shows summarized results from a complete simulation.

Table 5.3 Results of real time complicated dispatching strategies

Jobs	QC	YC	PM	Real time (min) (look ahead:2 jobs)	Real time (min) (look ahead:1 job)	Simple rule (min) (real time)
			15	51.30	53.31	61.55
50	10	8	20	50.41	53.24	61.55
			30	50.53	53.88	61.55
			15	57.73	60.04	67.51
60	10	8	20	55.79	65.25	68.32
			30	55.65	60.97	68.28
			15	61.59	64.80	74.27
70	10	8	20	59.96	65.21	74.31
			30	59.40	66.44	74.27
			15	67.84	73.21	83.54
80	10	8	20	66.83	72.52	84.14
			30	66.63	72.30	83.51
			15	78.56	82.02	91.47
90	10	8	20	76.35	81.05	91.55
			30	75.35	80.71	91.47
			15	87.69	90.32	100.24
100	10	8	20	85.20	89.05	101.01
			30	82.53	88.81	99.82

In this set of experiments, we compare 1 job look ahead and 2 jobs look ahead, as well as simple greedy rule simulation embodied in Microport respectively. We test different scenarios with job number ranging from 50-100, and PM number ranging from 15-30. From the results shown in Table 5.3, we can observe that in each instance, the makespan time obtained from 2 jobs look ahead method is always the minimal value, which is to state that uncertainties in real practice will undermine the per-deterministic model performance. The results of look 1 job ahead experiments are close to those of look 2 jobs

ahead method, while being better than simple greedy rule simulation results. From this table, we can observe that it is very important to absorb the real time information when making online scheduling. Sophisticated heuristic algorithms can improve the seaport terminal performances in the perfect experimental environment; however in real practice, these models cannot reach its optimal level due to real constraints. Thus we present this real time simulation platform to solve the online scheduling more efficiently. With this simulation platform, we can do better terminal planning in real time environment.

5.6 Summary

In real operations, terminal operators need an advanced IT platform to aid real time decision making. These techniques could provide real time status of vessels and containers, handle exceptions and modify plans rapidly, and improve overall planning efficiency and coordinate all terminal equipment. Terminals would benefit from these IT infrastructures by being efficient, flexible, cost-effective and scalable. In this chapter, different real time dispatching rules are evaluated using the proposed simulation platform. This platform is built to facilitate real time decision making with less human efforts. It has three main merits: firstly, it can flexibly communicate with external optimization models, which enables testing of different complicated heuristics; secondly, it can generate different layouts easily for testing various simulation scenarios; thirdly, it can aid real time dispatching because of its fast speed. This is critical to the running of any container terminal.

In this chapter, we test different real time dispatching rules in the proposed platform. Firstly, we test some simple rules which are embodied directly into Microport. Some simple greedy dispatching strategies have performed reasonably well (Most Critical QC strategy). We have shown that simple look-ahead dispatching strategy with the use of GRASP has outperformed all other strategies in terms of quay crane efficiency and yet maintaining a low level of vehicle waiting time proportion and high vehicle utilization. On the other hand, we test the performance of a complicated dispatching model (GA-MCF model) in our simulation platform to capture more information. The results illustrate the importance of capturing real time information in terminal scheduling.

6 Conclusion and Future Research

6.1 Conclusion

To improve the performance of container terminals, various methods have been proposed to increase the productivity of all kinds of equipment used in container terminals. However, most of existing literatures focus on optimizing one or two equipment in the container terminal rather than optimizing all equipment (YC, QC and PM) as a whole system. Therefore, the optimal management plan for all equipment used in the container terminal is needed to improve the performance of the container terminal. This is crucial to guarantee that the terminal system can react in the most cost-effective way to meet the continuous growth of container traffic. The objective of this thesis is to study the optimization of the integrated dispatching problem, especially for transshipment hubs, considering coordination of different equipment. In this thesis, we discuss the vehicle dispatching problem in an integrative view, to answer the question of how to assign the container delivery jobs to PMs as a way of synchronizing operations of various types of handling equipment in a container terminal.

Firstly, we address the integrated dispatching problem for transshipment hubs. In our problem, we consider the delay at the yard side, and also include both the loading and discharging jobs simultaneously, which are commonly found in the transshipment port. Moreover the PMs are pooled among all the QCs rather than dedicated to a certain QC. We seek to provide an efficient way of dispatching vehicles to minimize the makespan

time at the quay side for a given number of container jobs by considering all equipment. The reason we use this objective is that we want to speed up the vessel turnaround time. Other objectives like QC waiting time, vessel turnaround time can also be considered in our model.

We develop a mixed integer programming (MIP) model for this problem. Numerical experiments show that the existing solver cannot be used to solve this MIP directly. Hence we propose two heuristics to tackle this problem. The first approach is a neighborhood search based method. For the second method, we combine the genetic algorithm (GA) with the minimum cost flow (MCF) network model. In the GA approach, instead of representing the chromosome using PM job sequence directly, we represent it in terms of the ready times of the jobs, and then MCF is used to decode the chromosome to compute the PM job sequence. This approach is innovative and can exploit good properties found in both GA and MCF. Given the ready times, the MCF aims at finding the PM sequence that minimizes the objective for the MCF model. It is shown that there exists optimal ready times for jobs which MCF can decode the chromosome into the optimal PM job sequence. The GA has a good property in searching the design space both locally and globally if the chromosome representation has good neighborhood structure. In this case, we feel that representing the chromosome using ready times might be better than using job sequence because the neighborhood structure for ready times can be preserved when the crossover operation is performed. We have shown that by using ready times as the chromosome representation, we are able to keep the neighborhood structure and hence good solutions can be found by GA. Moreover, we have shown that

by choosing appropriate ready times, we can get the optimal PM sequence. In the numerical runs, we show the superiority of the GA-MCF method over the VNS.

Secondly, we discuss how to assign the container jobs to PMs as well as the assignment of discharging jobs to yard locations. From a systematic angle, we argue that the discharged container yard location is also an important decision variable which may affect the performance of the whole system. Thus we consider the whole dispatching process to get a seamless workflow and minimize the waiting time in both QC and YC in the transshipment context. Three methods are proposed to tackle this integrated problem and they are NP, BSG and BPG methods. We have shown that the NP method can solve this problem efficiently by using proper parameters; moreover, BSG can generate a feasible solution very quickly which can used in real time dispatching, while the improved BPG method dominates the other two methods in getting a good quality solution in a limited time.

Finally, an integrative simulation platform is presented for the real time dispatching problem. This platform is built to facilitate real time decision makings with less human efforts. It has three main merits: firstly, it can flexibly communicate with external optimization models, which enables testing of different complicated heuristics; secondly, it can generate different layouts easily for testing various simulation scenarios; thirdly, it can aid real time dispatching because of its fast speed. This is critical to the operation of any container terminal. In this platform, we can test different real time dispatching rules, not only for various simple rules but also for some complicated models. The results

illustrate the importance of capturing real time information in terminal scheduling. This integrative simulation platform can be used directly by the port operator for the evaluation purpose.

6.2 Future Research Topics

There are several topics related to the scope of this thesis where future research can be conducted.

Firstly, for the dispatching model, we do not consider the uncertainty in the input data. However, in practice there may be some randomness involved, especially in travel times of yard trucks and the yard crane and quay crane processing times of jobs. Uncertainty in the input data will definitely make the problem much more complex, and hence efficient methods to solve such stochastic models are potential areas of future research.

Secondly, this study is focused on single load vehicle dispatching. Further studies could be done to see whether the same set of algorithms would just be as applicable in the case where dual-loading is allowed. In the case of dual-loading, each vehicle will inherently have more states and that will further increase the complexity of the problem, especially in a look-ahead approach.

References

[1] Álvarez J.F. A heuristic for vessel planning in a reach stacker terminal. J Maritime Res 3(1), pp.3–16.2006.

[2] Álvarez J.F. A Lagrangian relaxation approach to the vessel planning problem. Working paper, Pompeu Fabra University, Barcelona, 2007.

[3] Ambrosino D, Sciomachen A, Tanfani E. A decomposition heuristics for the container ship stowage problem. J Heuristics 12,pp.211–233.2006.

[4] Avriel, M. and Penn, M. Exact and Approximate Solutions of the Container Ship Stowage Problem. Computers and Industrial Engineering 25(1-4), pp.271-274. 1993.

[5] Avriel, M., Penn, M., Shpirer, N., and Witteboon, S. Stowage Planning for Container Ships to Reduce the Number of Shifts. Annals of Operations Research 76, pp. 55-71. 1998.

[6] Avriel, M., Penn, M., and Shpirer, N. Container Ship Stowage Problem: Complexity and Connection to the Coloring of Circle Graphs. Discrete Applied Mathematics 103(1-3), 271-279. 2000.

[7] Bielli, M., Boulmakoul, A. & Rida, M. Object oriented model for container terminal distributed simulation. European Journal of Operational Research, 175(3), pp.1731-1751. 2006.

[8] Bish, E.K., Leong, T., Li, C.., Ng, J.W.C., and Simchi-Levi, D. Analysis of a New Vehicle Scheduling and Location Problem. Naval Research Logistics 48, pp.363-385. .2001.

[9] Bish, E.K. A Multiple-Crane-Constrained Scheduling Problem in a Container Terminal. European Journal of Operational Research 144, pp.83-107. 2003.

[10] Bish, E.K., Chen, F.Y., Leong, Y.T., Nelson, B.L., Ng, J.W.C., and Simchi-levi, D. Dispatching Vehicles in a Mega Container Terminal. OR Spectrum 27, pp.491-506. 2005.

[11] Böse, J., Reiners, T., Steenken, D., and Voβ, S. Vehicle Dispatching at Seaport Container Terminals Using Evolutionary Algorithms. In: Sprague R H (ed) Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, DTM-IT, pp.1-10. IEEE, Piscataway. 2000.

[12] Bostel, N. and Dejax, P. Models and Algorithms for Container Allocation Problems on Trains in a Rapid Transshipment Shunting Yard. Transportation Science 32 (4), pp.370-379. 1998.

[13] Briskorn, D., Drexl, A., and Hartmann, S. Inventory-based Dispatching of Automated Guided Vehicles on Container Terminals. OR Spectrum 28, pp.611-630. 2006.

[14] Bruzzone, A. G., Giribone, P. and Revetria, R. Operative requirements and advances for the new generation simulators in multimodal container terminals.Proceedings of the 1999 Winter Simulation Conference, pp.1243-1252.1999.

[15] Canonaco P, Legato P, Mazza RM, Musmanno R. A queuing network model for the management of berth crane operations. Computing & Operations Research 35(8), pp2432-2446. 2008.

[16] Carrascosa, C., Rebollo, M., Julian, V., and Botti, V. A MAS Approach for Port Container Terminal Management: the Transtainer Agent. In: Actas de SCI'01, pp.1-5. International Institute of Informatics and Systemics, Orlando, FL. 2001.

[17] Chan, S.H. Dynamic AGV-container Job Deployment Strategy. Master of Science, National University of Singapore. 2001.

[18] Chen, T. Yard Operations in the Container Terminal: a Study in the "Unproductive Moves". Maritime Policy & Management 26(1), pp.27-38. 1999.

[19] Chen, C.Y., Chao, S.L., and Hsieh, T.W. A Time-Space Network Model for the Space Resource Allocation Problem in Container Marine Transportation. In Proceedings of the 17th International Symposium on Mathematical Programming, 2000, Atlanta, USA. 2000.

[20] Chen, Y., Leong, Y.T., Ng, J.W.C., Demir, E.K., Nelson, B.L., and Simchi-Levi, D. Dispatching Automated Guided Vehicles in a Mega Container Terminal, paper presented at INFORMS Montreal, May 1998.

[21] Chen, L., Bostel, N., Dejax, P., Cai, J. and Xi, L. A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal, EJOR, 181(1), pp.40-58. 2007

[22] Cheng Y. L., Sen H. C., Natarajan K., Teo C. P. & Tan K. C. Dispatching Automated Guided Vehicles in a Container Terminal, Handbook on Supply Chain Optimization. 2005

[23] Chung, Y.G., Randhawa, S.U., and Mcdowell, E.D. A Simulation Analysis for a Transtainer-based Container Handling Facility. Computers and Industrial Engineering 14(2), pp.113-115. 1988.

[24] Cordeau, J.-F., M. Gendreau, A. Hertz, G. Laporte and J.-S. Sormany (2005). New heuristics for the vehicle routing problem, in: A. Langevin and D. Riopel (eds.), Logistics Systems: Design and Optimization, Kluwer, Boston.

[25] Daganzo, C.F. The Crane Scheduling Problem. Transportation Research B 23 (3), pp.159-175. 1989.

[26] De Castilho, B. and Daganzo, C.F. Handling Strategies for Import Containers at Marine Terminals. Transportation Research B 27 (2), pp.151-166. 1993.

[27] Dekker, R. Voogd, P., and Asperen, E. Advanced Methods for Container Stacking. OR Spectrum 28(4), pp.563-586. 2006.

[28] Demirci, E. Simulation Modelling and Analysis of a Port Investment. SIMULATION, 79(2), pp.94-105.2003

[29] Dubrovsky, O., Levitin, G., and Penn, M. A Genetic Algorithm with a Compact Solution Encoding for the Container Ship Stowage Problem. Journal of Heuristics 8, pp.585-599. 2002.

[30] Duinkerken, M.B., Evers, J.J.M., and Ottjes, J.A. TRACES: Traffic Control Engineering System: A Case Study on Container Terminal Automation. In the Proceedings of the 1999 Summer Computer Simulation Conference (SCSC 1999), July 1999, Chicago, USA. ISBN 1-56555-173-7.

[31] Duinkerken, M.B. and Ottjes, J.A. A Simulation Model for Automated Container Terminals. In the Proceedings of the Business and Industrial

Simulation Symposium (ASTC 2000), April 2000, Washington D.C., USA. ISBN 1-56555-199-0.

[32] Duinkerken, M.B., Evers, J.J.M., and Ottjes, J.A. A Simulation Model for Integrating Quay Transport and Stacking Policies on Automated Container Terminals. In the Proceedings of the 15th European Simulation Multi-conference (ESM 2001), June 2001, Prague, Czechoslovakia. ISBN 1-56555-225-3.

[33] Duinkerken, M.B., Evers, J.J.M., and Ottjes, J.A. Improving Quay Transport on Automated Container Terminals. In the Proceedings of the IASTED International Conference Applied Simulation and Modeling (ASM 2002), June 2002, Crete. ISBN 0-88986-334-2.

[34] Duinkerken, M.B., Dekker, R., Kurstjens, S.T.G.L., Ottjes, J.A., and Dellaert, N.P. Comparing Transportation Systems for Inter-Terminal Transport at the Maasvlakte Container Terminals. OR Spectrum 28, pp.469-493. 2006.

[35] Evers, J.J.M. and Koppers, S.A.J. Automated Guided Vehicle Traffic Control at a Container Terminal. Transportation Research A 30 (1), pp.21-34. 1996.

[36] Franz T, Voß S, Rölke H. Market-mechanisms for integrated container terminal management. In:Bertram V (ed) Proceedings of the 6th International Conference on Computer and IT Applications in the Maritime Industries, COMPIT'07, Cortona, April 23–25. INSEAN, pp 234–248.2007.

[37] Garcia-Lopez, F., Melian-Batista, B., Moreno-Perez, J.A. and Moreno-Vega, M. The parallel variable neighbourhood Search for the p-Median Problem, Journal of Heuristics, 8, pp.375-388. 2002.

[38] Gademann, A.J.R.M. and van de Velde, S.L. Positioning Automated Guided Vehicles in a Loop Layout. European Journal of Operational Research 127(3), pp.565-573. 2000.

[39] Gambardella, L.M., Rizzoli, A.E., and Zaffalon, M. Simulation and Planning of an Intermodal Container Terminal. Simulation 71 (2), pp.107-116. 1998.

[40] Gibson, R., Carpenter, B., and Seeburger, S. A Flexible Port Traffic Planning Model. In the Proceedings of the 1992 Winter Simulation Conference, pp.1296-1306. 1992.

[41] Giemsch, P., Jellinghaus, A. Organization Models for the Containership Stowage Problem. Paper presented at the Annual International Conference of the German Operations Research Society (OR 2003), Sep. 2003, Heidelberg, Germany.

[42] Goodchild A.V. Crane double cycling in container ports: algorithms, evaluation, and planning. Ph.D. thesis, University of California at Berkeley, Department of Civil and Environmental Engineering, California.2005.

[43] Goodchild AV (2006a) Crane double cycling in container ports: planning methods and evaluation. Paper presented at the METRANS National Urban Freight Conference, Long Beach, 1–3 February 2006. http://www.metrans.org/nuf/documents/Goodchild.pdf. Accessed 19 April 2007

[44] Goodchild A.V. Port planning for double cycling crane operations. Paper presented at the TRB 2006 annual meeting, Washington, DC, 22–26 January 2006b.

[45] Goodchild AV, Daganzo CF. Reducing ship turn-around time using double cycling. Technical report, University of California, Institute of Transportation Studies, Berkeley, Research report UCB-ITS-RR-2004-4, 1 May.2004.

[46] Goodchild AV, Daganzo CF. Crane double cycling in container ports: effect on ship dwell time. Technical report, University of California, Institute of Transportation Studies, Berkeley. Research Report UCB-ITS-RR-2005-5, July.2005.

[47] Goodchild AV, Daganzo CF. Double-cycling strategies for container ships and their effect on ship loading and unloading operations. Transport Sci 40,pp.473–483.2006

[48] Goodchild AV, Daganzo CF Crane double cycling in container ports: planning methods and evaluation.Transport Res B 41,pp.875–891.2007

[49] Grunow, M., Günther H.-O., and Lehmann, M. Dispatching Multi-Load AGVs in Highly Automated Seaport Container Terminals. OR Spectrum 26(2), pp.211-235. 2004.

[50] Grunow, M., Günther H.-O., and Lehmann, M. Strategies for Dispatching AGVs at Automated Seaport Container Terminals. OR Spectrum 28, pp.587-610. 2006.

[51] Guan, Y., Xiao, W.Q., Cheung, R.K., and Li, C.L. A Multiprocessor Task Scheduling Model for Berth Allocation: Heuristic and Worst Case Analysis. Operations Research Letters 30, 343-350. 2002.

[52] Guan, Y., Cheung, R.K. The Berth Allocation Problem: Models and Solution Methods. OR Spectrum 26, pp.75-92. 2004

[53] Ha, B.H., Park, E.J. & Lee, C.H. A simulation model with a low level of detail for container terminals and its applications. In Proceedings of the 39th Conference on Winter Simulation: 40 years! The best is yet to come. pp. 2003–2011. 200

[54] Hansen, P. and N. Mladenovic. Variable Neighborhood Search for the p-Median. Location Science 5, pp.207-226.1997

[55] Hartmann, S. A General Framework for Scheduling Equipment and Manpower on Container Terminals. OR Spectrum 26, pp.51-74. 2004.

[56] Hartmann, S. Generating Scenarios for Simulation and Optimization of Container Terminal Logistics. OR Spectrum 26(2), pp.171-192. 2004.

[57] Hayuth, Y., Pollatschek, M.A., Roll, Y. Building a Port Simulator. Simulation 63, pp.179-189. 1994.

[58] Henesey, L., Wernstedt, F., Davidsson, P. A Market Based Approach to Container Port Terminal Management. In: Proceedings of the 15th European Conference on Artificial Intelligence, Workshop – Agent Technologies in Logistics, Lyon, France, 2002.

[59] Henesey L, Davidsson P, Persson JA Using simulation in evaluating berth allocation at a container terminal. Paper presented at the Third International EuroConference on Computer Applications and Iinformation Technology in the Maritime Industries, COMPIT'04, Siguënza, 9–12 May 2004.

[60] Henesey L, Davidsson P, Persson JA Agent based simulation architecture for evaluating operational policies in transshipping containers. In: Multiagent System Technologies, Proceedings of Fourth German Conference, MATES 2006 Erfurt, 19–20 September 2006. Springer, Berlin, pp 73–85.2006.

[61] Henesey L, Wernstedt F, Davidsson P Market-driven control in container terminal management. Paper presented at the second international EuroConference on Computer Applications and Information Technology in the Maritime Industries, COMPIT'03, Hamburg, Germany, 14–17 May 2003

[62] Henesey LE (2006) Multi-agent systems for container terminal management. Blekinge Institute of Technology Doctoral Dissertation Series. Blekinge Institute of Technology, Karlshamn.

[63] Hirashima Y, Takeda K, Harada S, DengM, Inoue A. A Q-learning for group-based plan of container transfer scheduling. JSME Int J Series C 49,pp.473–479.2006

[64] Holgu ń-Veras, J. and Walton, C. On the Development of a Computer System to Simulate Port Operations Considering Priorities. In: Proceedings of the 28th Conference on Winter Simulation, pp. 1471-1478. ACM Press, New York. 1996.

[65] Holgu ń-Veras, J. and Jara-D áz, S. Optimal Pricing for Priority Service and Space Allocation in Container Ports. Transportation Research B 33, pp.81-106. 1999.

[66] Hulten, L.A.R. Container Logistics and its Management. PhD thesis, Chalmers University of Technology: Department of Transportation and Logistics. 1997.

[67] Imai, A., Nagaiwa, K., and Tat, C.W. Efficient Planning of Berth Allocation for Container Terminals in Asia. Journal of Advanced Transportation 31 (1), pp.75-94. 1997.

[68] Imai, A., Nishimura, E., and Papadimitriou, S. The Dynamic Berth Allocation Problem for a Container Port. Transportation Research-B 35(4), pp.401-417. 2001.

[69] Imai, A., Nishimura, E., and Papadimitriou, S. Berth Allocation with Service Priority. Transportation Research-B 37(5), pp.437-457. 2003.

[70] Imai A, Sasaki K, Nishimura E, Papadimitriou S . Multi-objective simultaneous stowage and load planning for a container ship with container rehandle in yard stacks. Eur J Oper Res 171,pp.373–389.2006.

[71] Imai A, Nishimura E, Current J. A Lagrangian relaxation-based heuristic for the vehicle routing with full container load. Eur J Oper Res 176, pp.87–105.2007a.

[72] Imai A, Nishimura E, Hattori M, Papadimitriou S. Berth allocation at indented berths for mega containerships. Eur J Oper Res 179, pp.579–593.2007b.

[73] Johansen RS. Container terminal planning: improving system productivity to service larger container vessels. Port Technol Int V31, pp104-106.2006.

[74] Kang, J.-G. and Kim, Y.-D. Stowage Planning in Maritime Container Transportation. Journal of the Operational Research Society 53, pp.415-426. 2002.

[75] Kang J, Ryu KR, Kim KH Deriving stacking strategies for export containers with uncertain weight information. J Intell Manufact 17, pp.399–410.2006a.

[76] Kang J, Ryu KR, Kim KH Determination of storage locations for incoming containers of uncertain weight. In: Ali M, Dapoigny R (eds) Advances in Applied Artificial Intelligence, Proceedings of the 19th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2006, Annecy, June 27–30. Springer, Berlin, pp 1159–1168.2006b.

[77] Kia, M., Shayan, E., and Ghotb, F. Investigation of Port Capacity Under a New Approach by Computer Simulation. Computers & Industrial Engineering 42, pp.533-540. 2002.

[78] Kim, K.H. Evaluation of the Number of Reshuffles in Storage Yards. Computers and Industrial Engineering 32(4), pp.701-711. 1997.

[79] Kim, K.H. and Bae, J.W. Re-marshaling Export Containers in Port Container Terminals. Computers & Industrial Engineering 35, pp.655-658. 1998.

[80] Kim, K.H. and Bae, J.W. A Dispatching Method for Automated Guided Vehicles to Minimize Delays of Containership Operations. International Journal of Management Science 5 (1), pp.1-25. 1999.

[81] Kim, K.H. and Kim, H.B. Segregating Space Allocation Models for Container Inventories in Port Container Terminals. International Journal of Production Economics 59, pp.415-423. 1999a.

[82] Kim, K.H. and Kim, K.Y. Routing Straddle Carriers for the Loading Operation of Containers Using a Beam Search Algorithm. Computers & Industrial Engineering 36 (1), pp.109-136. 1999b.

[83] Kim, K.Y. and Kim, K.H. A Routing Algorithm for a Single Straddle Carrier to Load Export Containers onto a Containership. International Journal of Production Economics 59 (1-3), pp.425-433. 1999c.

[84] Kim K.H. and Kim K. Y. Optimal price scheduling for storage of inbound containers. Transport Research-B 41, pp. 892-905.2007.

[85] Kim, K.H. and Moon, K.C. Berth Scheduling by Simulated Annealing. Transportation Research-B 37(6), pp.541-560. 2003.

[86] Kim, K.H., Park, Y.M., and Ryu, K.R. Deriving Decision Rules to Locate Export Containers in Storage Yards. European Journal of Operational Research 124, pp.89-101. 2000.

[87] Kim K.H. and Park Y.M. A crane scheduling method for port container terminals. Eur J Oper Res 256, pp. 752-768.2004.

[88] Kim, K.H., Wang, S.J., Park, Y.M., Yang, C.H., and Bae, J.W. A Simulation Study on Operation Rules for Automated Container Yards. Presentation/Proc. of the 7th Annual International Conference on Industrial Engineering. 2002.

[89] Kim KH,Won SH, Lim JK, Takahashi T. An architectural design of control software for automated container terminals. Comput Ind Eng 46:741–754.2004.

[90] Kim YH, Park T, RyuKR. Dynamic weight adjustment for developing a stacking policy for automated container terminals. Paper presented at the International Conference on Intelligent Manufacturing and Logistics Systems (IML 2007), Kitakyushu, Japan, 26–28 February 2007.

[91] Kim, K.H., Lee, K.M., and Hwang, H. Sequencing Delivery and Receiving Operations for Yard Cranes in Port Container Terminals. International Journal of Production Economics 84(3), pp.283-292. 2003.

[92] Koh, P.-H., Goh, J.L.K., Ng, H.-S., and Ng, H.-C. Using Simulation to Preview Plans of Container Port Operations. In the Proceedings of the 26th Conference on Winter Simulation, pp.1109-1115. Society for Computer Simulation International, San Diego, CA. 1994.

[93] Konings, J.W. Integrated Centres for the Transshipment, Storage, Collection and Distribution of Goods: a Survey of the Possibilities for a High-Quality Intermodal Transport Concept. Transport Policy 3(1-2), pp.3-11. 1996.

[94] Koo, P.H., Lee, W.S., and Jang, D.W. Fleet Sizing and Vehicle Routing for Container Transportation in a Static Environment. OR Spectrum 26(2), pp.193-209. 2004.

[95] Kozan, E. Comparison of Analytical and Simulation Planning Models of Seaport Container Terminals. Transportation Planning and Technology 20 (3), pp.235-248. 1997.

[96] Kozan, E. and Preston, P. Genetic Algorithms to Schedule Container Transfers at Multimodal Terminals. International Transactions in Operational Research 6 (3), pp.311-329. 1999.

[97] Kozan, E. and Preston, P. Mathematical Modeling of Container Transfers and Storage Locations at Seaport Terminals. OR Spectrum 28(4), pp.519-537. 2006.

[98] Lau HYK, Wong VWK, Lee ISK Immunity-based autonomous guided vehicles control. Appl Soft Comput 7,pp.41–57.2007.

[99] Lee, T., Park, N. & Lee, D.. A simulation study for the logistics planning of a container terminal in view of SCM - PB - Routledge. Maritime Policy & Management: The flagship journal of international shipping and port research, 30(3), pp.243. 2003.

[100] Lee, L.H., Chew, E.P., Tan, K.C., and Han Y. An Optimization Model for Storage Yard Management in Transshipment Hubs. OR Spectrum 28(4), pp.539-561. 2006.

[101] Lee, L.H. et al. A simulation study on the uses of shuttle carriers in the container yard. In Proceedings of the 39th Conference on Winter Simulation: 40 years! The best is yet to come. pp. 1994–2002. 2007.

[102] Lee, L.H. et al. A study on port design automation concept. In Proceedings of the 40th Conference on Winter Simulation. pp. 2726–2731. 2008.

[103] Lee, L.H. et al. Vehicle Dispatching Algorithm for Transshipment Hubs. *OR Spectrum*, 32 (3), pp. 663-685.2010.

[104] Leeper, J.H. Integrated Automated Terminal Operations. Transportation Research Circular 33 (2), pp.23-28. 1988.

[105] Lehmann, M., Grunow, M., and Günther H.-O. Deadlock Handling for Real-Time Control of AGVs at Automated Container Terminals. OR Spectrum 28, pp.631-657. 2006.

[106] Leong, C.Y. Simulation Study of Dynamic AGV-Container Job Deployment Scheme. Master of Science, National University of Singapore. 2001.

[107] Liang C, Mi W. A quay crane scheduling problem by hybrid evolutionary algorithm for berth allocation planning. Paper presented at the International Conference on Intelligent Manufacturing and Logistics Systems (IML 2007), Kitakyushu, Japan, 26–28 February 2007.

[108] Li, C.L. and Vairaktarakis, G.L. Loading and Unloading Operations in Container Terminals. IIE Transactions 36, pp.287-297. 2004.

[109] Lim, J.K., Kim, K.H., Yoshimoto, K., and Lee, J.H. A Dispatching Method for Automated Guided Vehicles by Using a Bidding Concept. OR Spectrum 25, pp.25-44. 2003.

[110] Lim A, Xu Z. A critical-shaking neighborhood search for the yard allocation problem. Eur J Oper Res 174, pp.1247–1259.2006.

[111] Linn R, Liu J, Wan Y-w, Zhang C. Predicting the performance of container terminal operations using artificial neural networks. In: Bichou K, Bell MGH,

Evans A (eds) Risk management in port operations, logistics and supply chain security, Chap 8. Informa, London. 2007.

[112] Liu, C.I., Jula, H., Ioannou, P.A. Design, Simulation, and Evaluation of Automated Container Terminals. IEEE Transactions on Intelligent Transportation Systems 3(1), pp.12-26. 2002.

[113] Liu, C.I., Jula, H., Vukadinovic, K., and Ioannou, P. Automated Guided Vehicles System for Two Container Yard Layouts. Transportation Research Part C 12, pp.349-368. 2004.

[114] Mattfeld, D. The Management of Transshipment Terminals. Habilitation thesis, University at Bremen. 2003.

[115] Meersmans, P.J.M. Optimization of Container Handling Systems. Thela Thesis, Amsterdam. 2002.

[116] Meersmans, P.J.M. and Wagelmans, A.P.M Effective Algorithms for Integrated Scheduling of Handling Equipment at Automated Container Terminals. Technical Report EI 2001-19, Erasmus University Rotterdam, Econometric Institute. 2001a.

[117] Meersmans, P.J.M. and Wagelmans, A.P.M. Dynamic Scheduling of Handling Equipment at Automated Container Terminals. Technical Report EI 2001-33, Erasmus University Rotterdam, Econometric Institute, 2001b.

[118] Merkuryev, Y., Tolujew, J., Blümel, E., Novitsky, L., Ginters, E., Viktorova, E., Merkuryeva, G., and Pronins, J. A Modeling and Simulation Methodology for Managing the Rigaharbour Container Terminal. Simulation 71, pp.84-95. 1998.

[119] Moorthy, R. and Teo, C.-P. Berth Management in Container Terminal: the Template Design Problem. OR Spectrum 28, pp.495-518. 2006.

[120] Moorthy, R.L., Hock-Guan, W., Wing-Cheong, N., and Chung-Piaw, T. Cyclic Deadlock Prediction and Avoidance for Zone-Controlled AGV System. International Journal of Production Economics 83(3), pp.309-324. 2003.

[121] Mosca, R., Giribone, P., Bruzzone, A. Simulation and Automatic Parking in a Training System for Container Terminal Yard Management. In: Proceedings of ITEC, pp.65-72. 1994.

[122] Murty, K.G., Liu, J., Wan, Y., and Linn, R. A Decision Support System for Operations in a Container Terminal. Decision Support System 39(3), pp.309-332. 2005.

[123] Murty, K.G., Wan, Y., Liu, J., Tseng, M.M., Leung, E., Lai, K.K., Chiu, H.W.C. Hong Kong International Terminals Gains Elastic Capacity Using a Data-Intensive Decision-Support System. In the Proceedings of INFORMS 2005, Jan. 2005, pp.61-75.

[124] Ng WC, Mak KL Sequencing of container pickup trucks in container yard blocks. Int J Ind Eng 11, pp.82–89.2004.

[125] Ng WC, Mak KL, Zhang YX Scheduling trucks in container terminals using a genetic algorithm. Eng Optimization 39:33–47.2007.

[126] Nguyen VD, Kim KH. A dispatching method for automated lifting vehicles in automated port container terminals. Paper presented at the International Conference on Intelligent Manufacturing and Logistics Systems (IML 2007), Kitakyushu, Japan, 26–28 February 2007.

[127] Nishimura E, Imai A, Papadimitriou S. Yard trailer routing at a maritime container terminal. Transport Res E 41, pp.53–76.2005.

[128] Nam, K.-C., Ha, W.-I. Evaluation of Handling Systems for Container Terminals. Journal of Waterway, Port, Coastal and Ocean Engineering 127(3), pp.171-175. 2001.

[129] Nam, K., Kwak, K. & Yu, M. Simulation Study of Container Terminal Performance. Journal of Waterway, Port, Coastal, and Ocean Engineering, 128(3), pp.126-132. 2002

[130] Narasimhan, A. and Palekar, U.S. Analysis and Algorithms for the Transtainer Routing Problem in Container Port Operations. Transportation Science 36(1), pp.63-78. 2002.

[131] Nevins, M.R., Macal, C.M., Love, R.J., and Bragen, M.J. Simulation, Animation and Visualization of Seaport Operations. Simulation 71, pp.96-106. 1998a.

[132] Nevins, M.R., Macal, C.M. & Joines, J.C. A Discrete-Event Simulation Model for Seaport Operations. SIMULATION, 70(4), pp.213-223. 1998b.

[133] Nishimura, E., Imai, A., and Papadimitriou, S. Berth Allocation Planning in the Public Berth System by Genetic Algorithms. European Journal of Operational Research 131, pp.282-292. 2001.

[134] Nishimura, E., Imai, A., and Papadimitriou, S. Yard Trailer Routing at a Maritime Container Terminal. Transportation Research Part E 41, pp.53-76. 2005.

[135] Ottjes, J.A., Veeke, H.P.M., Duinkerken, M.B., Rijsenbrij, J.C., and Lodewijks, G. Simulation of a Multi-terminal System for Container Handling. OR Spectrum 28, pp.447-468, 2006.

[136] Park, K.T., Kim, K.H. Berth Scheduling for Container Terminals by Using a Subgradient Optimization Technique. Journal of the Operational Research Society 53, pp.1054-1062. 2002.

[137] Park, Y.-M., Kim, K.H. A Scheduling Method for Berth and Quay Cranes. OR Spectrum 25, pp.1-23. 2003.

[138] Peterkofsky, R.I. and Daganzo, C.F. A Branch and Bound Solution Method for the Crane Scheduling Problem. Transportation Research B 24 (3), pp.159-172. 1990.

[139] Petering, M., 2007. Design, analysis, and real-time control of seaport container transshipment terminals. University of Michigan. 2007.

[140] Potvin, J.Y., Y. Shen, J.M. Rousseau. Neural networks for automated vehicle dispatching. Computers and Operations Research, 19, pp.267-276.1992

[141] Potvin, J.Y., G. Dufour, J.M. Rousseau. Learning vehicle dispatching with linear programming models. Computers and Operations Research, 20(4), pp.371-380.1993

[142] Preston, P. and Kozan, E. An Approach to Determine Storage Locations of Containers at Seaport Terminals. Computers & Operations Research 28, pp.983-995. 2001.

[143] Ramani, K.V. An Interactive Simulation Model for the Logistics Planning of Container Operations in Seaports. Simulation 66 (5), pp.291-300. 1996.

[144] Rebollo, M., Julian, V., Carrascosa, C., and Botti, V. A MAS Approach for Port Container Terminal Management. In: Proceedings of the 3rd Iberoamerican Workshop on DAI and MAS, 2000, pp.83-94. Atibaia, Sao Paulo, Brasil. 2000.

[145] Reveliotis, S.A. Conflict Resolution in AGV Systems. IIE Transactions 32, pp.647-659. 2000.

[146] Rizzoli, A.E., Gambardella, L.M., Zaffalon, M., and Mastrolilli, M. Simulation for the Evaluation of Optimised Operations Policies in a Container Terminal. In: HMS99, Maritime & Industrial Logistics Modeling and Simulation, Sep. 1999, Genoa, Italy.

[147] Roach, P.A. and Wilson, I.D. A Genetic Algorithm Approach to Strategic Stowage Planning for Container-Ships. Working paper, University of Glamorgan, UK. 2002.

[148] Robert Stahlbock, Stefan Voß. Operations research at container terminals: a literature update. OR Spectrum 30(1): pp.1-52. 2008.

[149] Saanen, Y.A. Examining the Potential for Adapting Simulation Software to Enable Short-Term Tactical Decision Making for Operational Optimisation. Technical Report, TBA Nederland/Delft University of Technology. 2000.

[150] Schneidereit, G., VoβS, Parra, A., and Steenken, D. A General Pickup and Delivery Problem for Automated Guided Vehicles with Multi-loads: a Case Study. Working paper, University of Hamburg. 2003.

[151] Sculli, D., Hui, C.F. Three Dimensional Stacking of Containers. Omega 16(6): pp.585-594. 1988.

[152] Shi.L.and S. Ólafsson. An Integrated Framework for Deterministic and Stochastic Optimization, in S. Andradóttir, K.J. Healy, D.H. Withers, and B.L. Nelson (eds.), Proceedings of the 1997 Winter Simulation Conference, pp.358-365.1997.

[153] Shi, L., Olafsson, S., and Sun, N. A New Parallel Randomized Algorithm for Traveling Salesman Problem. Computer and Operations Research.26. pp. 371-394.2000.

[154] Sgouridis, S.P., Makris, D., Angelides D.C. Simulation Analysis for Midterm Yard Planning in Container Terminal. Journal of Waterway, Port, Coastal and Ocean Engineering, pp.178-187. 2003.

[155] Shabayek, A.A., Yeung, W.W. A Simulation for the Kwai Chung Container Terminal in Hong Kong. European Journal of Operational Research 40, pp.1-11. 2002.

[156] Steenken, D. Fahrwegoptimierung am Containerterminal under Echtzeitbedingungen. OR Spectrum 14 (3), pp.161-168. 1992.

[157] Steenken, D., Henning, A., Freigang, S., and Voβs, S. Routing of Straddle Carriers at a Container Terminal with the Special Aspect of Internal Moves. OR Spectrum 15 (3), pp.167-172. 1993.

[158] Steenken, D., Winter, T., and Zimmermann, U.T. Stowage and Transport Optimization in Ship Planning. In: Grötschel M, Krumke S O, Rambau J (eds) Online optimization of large scale systems, pp 731–745. Springer, Berlin. 2001.

[159] Steenken, D., Voβ, S., and Stahlbock, R. Container Terminal Operation and Operations Research - a Classification and Literature Review. OR Spectrum 26, 3-49, 2004.

[160] Tavakkoli-Moghaddam, R., & Safaei, N. An evolutionary algorithm for a single-item resource-constrained aggregate production planning problem. In IEEE Congress on Evolutionary Computation, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, July 16–21. Canada.2006.

[161] Taleb-Ibrahimi, M., De Castilho, B., and Daganzo C.F. Storage Space vs Handling Work in Container Terminals. Transportation Research B 27, pp.13-32. 1993.

[162] Thiers, G.F. and Janssens, G.K. A Port Simulation Model as a Permanent Decision Instrument. Simulation 71, pp.117-125. 1998.

[163] Van der Meer, R. Operational Control of Internal Transport, ERIM Ph.D. series Research in Management 1. 2000.

[164] Veeke, H.P.M. and Ottjes, J.A. Detailed Simulation of the Container Flows for the IPSI Concept. In the Proceedings of the 11th European Simulation Symposium (ESS 1999), October 1999, Erlangen.

[165] Veeke, H.P.M., Ottjes, J.A. A Generic Simulation Model for Systems of Container Terminals. In the Proceedings of the 16th European Simulation Multiconference (ESM 2002), June 2002, Darmstadt, Germany.

[166] Vis, I.F.A., Harika, I. Comparison of Vehicle Types at an Automated Container Terminal. OR Spectrum 26, pp.117-143. 2004.

[167] Vis, I.F.A., de Koster, R. Transshipment of Containers at a Container Terminal: an Overview. European Journal of Operational Research 147, pp.1-16. 2003.

[168] Vis, I.F.A., Koster, R., de Roodbergen, K.J., and Peeters, L.W.P. Determination of the Number of Automated Guided Vehicles Required at a Semi-Automated Container Terminal. Journal of the Operational Research Society 52, pp.409-417. 2001.

[169] Wilson, I.D., Roach, P.A. Principles of Combinatorial Optimization Applied to Container-Ship Stowage Planning. Journal of Heuristics 5, pp.403-418. 1999.

[170] Wilson, I.D. and Roach, P.A. Container Stowage Planning: a Methodology for Generating Computerized Solutions. Journal of the Operational Research Society 51, pp.1248-1255. 2000.

[171] Wilson, I.D., Roach, P.A., and Ware, J.A. Container Stowage Pre-planning: Using Search to Generate Solutions, a Case Study. Knowledge-Based Systems 14(3-4): pp.137-145. 2001.

[172] Yang, C.H., Choi, Y.S., and Ha, T.Y. Performance Evaluation of Transport Vehicle at Automated Container Terminal Using Simulation. OR Spectrum 26(2), pp. 149-170. 2004.

[173] Yun, W.Y. and Choi, Y.S. A Simulation Model for Container-Terminal Operation Analysis Using an Object-Oriented Approach. International Journal of Production Economics 59 (1-3), pp.221-230. 1999.

[174] Yun, W.Y., Choi, Y.S. Simulator for Port Container Terminal Using an Object Oriented Approach. Working paper, Pusan National University. 2003.

[175] Zaffalon, M., Rizzoli, A.E., Gambardella, L.M., and Mastrolilli, M. Resource Allocation and Scheduling of Operations in an Intermodal Terminal. In the Proceedings of 10th European Simulation Symposium and Exhibition, Simulation in Industry, pp.520-528, Oct. 1998, Nottingham, United Kingdom.

[176] Zhang, C., Liu, J., Wan, Y.W., Murty, K.G., and Linn, R.J. Storage Space Allocation in Container Terminals. Transportation Research B 37, pp.883-903. 2003.

[177] Zhang L-W, Ye R, Huang S-Y, Hsu W-J. (2005). Mixed integer programming models for dispatching vehicles at a container terminal. J Appl Math Comput 17,pp.145–170.2005